Deep Learning for Cyber Security Intrusion Detection: Approaches, Datasets, and Comparative Study

Mohamed Amine Ferrag, Leandros Maglaras, Sotiris Moschoyiannis, and Helge Janicke

***Abstract*—In this paper, we present a survey of deep learning approaches for cyber security intrusion detection, the datasets used, and a comparative study. Specifically, we provide a review of intrusion detection systems based on deep learning approaches. The dataset plays an important role in intrusion detection, therefore we describe 35 well-known cyber datasets and provide a classification of these datasets into seven categories; namely, network traffic-based dataset, electrical network-based dataset, internet traffic-based dataset, virtual private network-based dataset, android apps-based dataset, IoT traffic-based dataset, and internet-connected devices-based dataset. We analyze seven deep learning models including recurrent neural networks, deep neural networks, restricted Boltzmann machines, deep belief networks, convolutional neural networks, deep Boltzmann machines, and deep autoencoders. For each model, we study the performance in two categories of classification (binary and multiclass) under two new real traffic datasets, namely, the CSE-CIC-IDS2018 dataset and the Bot-IoT dataset. In addition, we use the most important performance indicators, namely, accuracy, false alarm rate, and detection rate for evaluating the efficiency of several methods.**

***Index Terms*—Machine Learning, Deep Learning, Cyber Security, Intrusion detection.**

I. Introduction

Critical National Infrastructures (CNIs) such as ports, water and gas distributors, hospitals, energy providers are becoming the main targets of cyber attacks. Supervisory Control and Data Acquisitions (SCADA) or Industrial Control Systems (ICS) in general are the core systems that CNIs rely on in order to manage their production. Protection of ICSs and CNIs has become an essential issue to be considered in an organizational, national and European level. For instance, in order to cope with the increasing risk of CNIs, Europe has issued during the past years a number of directives and regulations that try to create a coherent framework for securing networks, information and electronic communications. Apart from regulations, directives and policies, specific security measures are also needed to cover all legal, organizational, capacity building and technical aspects of cyber security [1].

(Corresponding author: Mohamed Amine Ferrag)

M. A. Ferrag is with the Department of Computer Science, Guelma University, Guelma 24000, Algeria email: (ferrag.mohamedamine@univ-guelma.dz)

L. Maglaras is with the School of Computer Science and Informatics, De Montfort University, Leicester U.K. (e-mail: leandros.maglaras@dmu.ac.uk)

S. Moschoyiannis is with the Department of Computer Science, University of Surrey, U.K. (e-mail: s.moschoyiannis@surrey.ac.uk)

H. Janicke is with the School of Computer Science and Informatics, De Montfort University, Leicester U.K. (e-mail: heljanic@dmu.ac.uk).

Intrusion detection systems (IDS) [2] are part of the second defense line of a system. IDSs can be deployed along with other security measures, such as access control, authentication mechanisms and encryption techniques in order to better secure the systems against cyber attacks. Using patterns of benign traffic or normal behavior or specific rules that describe a specific attack, IDSs can distinguish between normal and malicious actions [3]. According to [4], data mining which is used to describe knowledge discovery can help to implement and deploy IDSs with higher accuracy and robust behavior as compared to traditional IDSs that may not be as effective against modern sophisticated cyber attacks [5].

Moreover, many researchers are struggling to find comprehensive and valid datasets to test and evaluate their proposed techniques and having a suitable dataset is a significant challenge in itself. In order to test the efficiency of such mechanisms, reliable datasets are needed that (i) contain both benign and several attacks, (ii) meet real world criteria, and (iii) are publicly available [6]. This paper extends our work in [7].

Our contributions in this work are:

- We review the intrusion detection systems that use deep learning approaches.
- We present 35 well-known cyber datasets and provide a classification of these datasets into seven categories: network traffic-based dataset, electrical network-based dataset, internet traffic-based dataset, virtual private network-based dataset, android apps-based dataset, IoT traffic-based dataset, and internet-connected devices-based dataset.
- We analyze seven deep learning approaches according to two models, namely, deep discriminative models and generative/unsupervised models. The deep discriminative models include three approaches: (i) recurrent neural networks, (ii) deep neural networks, and (iii) convolutional neural networks. The generative/unsupervised models include four approaches: (i) deep autoencoders, (ii) restricted Boltzmann machine, and (iii) deep Boltzmann machines, and (iv) deep belief networks.
- We study the performance of each deep learning model using two new real traffic datasets, namely, the CSE-CIC-IDS2018 dataset and the Bot-IoT dataset.
- We compare the performance of deep learning approaches with four machine learning approaches, namely, Naive Bayes, Artificial neural network, Support Vector Machine, and Random forests.

TABLE I: Related studies on cyber security intrusion detection

Study	Year	DL	ML and DM	EDL	EML	Dsets
Buczak et al. [8]	2015	No	Yes	No	No	Yes
Milenkoski et al. [9]	2015	No	Partial	No	Partial	No
Folino et al. [10]	2016	No	Yes	No	No	Partial
Zarpelao et al. [11]	2017	No	Partial	No	No	No
Aburomman and Reaz [12]	2017	No	Yes	No	Partial	Partial
Xin et al. [13]	2018	Yes	Partial	No	No	Partial
Ring et ak. [14]	2019	No	No	No	No	Yes
Loukas et al. [15]	2019	No	No	No	No	Partial
da Costa et al. [16]	2019	No	No	No	No	Partial
Chaabouni et al. [17]	2019	Partial	Yes	No	Partial	Partial
Berman et al. [18]	2019	Yes	Partial	No	No	Partial
Mahdavifar et al. [19]	2019	Yes	Partial	No	No	Partial
Sultana et al. [20]	2019	No	Yes	No	No	No
Our Study	/	Yes	Partial	Yes	Yes	Yes

ML and DM: Machine learning (ML) and data mining (DM) approaches; DL: Deep learning approaches; EDL: Evaluation of deep learning approaches; EML: Evaluation of machine learning approaches; Dsets: A review of datasets used by IDSs.

The remainder of the paper is organized as follows. In Section II, we provide an overview of related studies. Section III gives the intrusion detection systems based on deep learning approaches. In Section IV, we present the different datasets used by deep learning approaches papers applied to intrusion detection. In Section V, we present seven deep learning approaches. In Section VI, we study the performance of each deep learning approach in binary classification and multiclass classification. Lastly, Section VII presents conclusions.

II. Related studies

In the literature, there are different related studies that deal with machine learning techniques for intrusion detection systems. As illustrated in Table I, we categorize the studies based on the following criteria:

- Deep learning approaches: it specifies if the study was focused on Deep learning approaches for intrusion detection systems.
- Machine learning approaches: it indicates whether the study considered machine learning approaches for intrusion detection systems.
- Evaluation of deep learning approaches: it indicates whether the study evaluates deep learning approaches for intrusion detection systems.
- Evaluation of machine learning approaches: it indicates whether the study evaluates machine learning approaches for intrusion detection systems.
- Datasets used by IDSs: it indicates whether the study focused on the datasets used for intrusion detection systems.

Recently, Ring et al. [14] presented a study of intrusion detection datasets. Specifically, the study presents 34 datasets and identifies 15 characteristics for them. These characteristics are categorized into five groups, namely, 1) General Information, 2) Evaluation, 3) Recording Environment, 4) Data Volume, 5) Nature of the Data, and General Information. Buczak et al. [8] presented a study of machine learning approaches used by the intrusion detection systems. This study classified the datasets into three types, namely, 1) packet-level data, 2) netflow data, and 3) public datasets. In addition, the study provided a computational complexity (i.e., time complexity) for each mining and machine learning approache used by the intrusion detection system. Zarpelao et al. [11] provided a comparative study of intrusion detection approaches in the internet of things (IoT). The study classified IDSs for IoT based on the detection technique, IDS placement technique, and security threat. Milenkoski et al. [9] provided the common practices in cyber security intrusion detection by analyzing existing systems related to each of the standard evaluation parameters, namely, workloads, metrics, and technique. Our study and four works [13], [18], [19], [20] focus on deep learning approaches that are designed for cyber security intrusion detection. However, these works do not give a comparative study of deep learning approaches on the datasets. To the best of our knowledge, our study is the first that thoroughly covers approaches, datasets, and a comparative study of deep learning for intrusion detection systems.

III. Deep learning approaches-based intrusion detection systems

This section describes the Deep learning approaches-based intrusion detection systems. As presented in Fig. 1, there are ten deep learning approaches used for cyber security intrusion detection, namely, 1) deep neural network, 2) feed forward deep neural network, 3) recurrent neural network, 4) convolutional neural network, 5) restricted Boltzmann machine, 6) deep belief network, 7) deep auto-encoder, 8) deep migration learning, 9) self-taught learning, and 10) replicator neural network.

A. Deep neural network

Tang et al. [21] proposed an intrusion detection system that employs a deep learning technique in software-defined networking. The proposed IDS system is implemented in the SDN controller which can monitor all the OpenFlow switches. The study used the NSL-KDD dataset under 2-class classification (i.e., normal and anomaly class), where the dataset consisted of four categories, namely, 1) DoS attacks, 2) R2L attacks, 3) U2R attacks, and 4) Probe attacks. The experimental results reported that the learning rate of 0.001 performed more effectively than others with the highest receiver operating characteristic curve (AUC). Potluri et al. [22] used the deep neural approach as the deep-category

TABLE II: Deep learning approaches for intrusion detection and dataset they use

Deep Learning Approach	IDS	Dataset Used	Performance metrics	Cited*
Deep neural network	Tang et al. (2016) [21]	NSL-KDD dataset	Precision, Recall, F1-score, Accuracy, ROC Curve	115
Deep neural network	Potluri et al. (2016) [22]	NSL-KDD dataset	Accuracy	40
Deep neural network	Kang et al. (2016) [23]	Vehicular network communication	FAR, ROC Curve, Detection Ratio	144
Deep neural network	Zhou et al. (2018) [24]	DOS, R2L, U2R, and PROBING	Accuracy, TPR, FPR	0
Deep neural network	Feng et al. (2019) [25]	KDD Cup 1999 dataset	Accuracy, Precision, Recall, F1-score	1
Deep neural network	Zhang et al. (2019) [26]	KDD Cup 1999 dataset	Accuracy, Precision, Recall, F1-Score	0
Deep neural network	Roy et al. (2017) [27]	KDD Cup 1999 dataset	Accuracy, Error	27
Deep neural network	Kim et al. (2017) [28]	KDD Cup 1999 dataset	Accuracy, detection rate, false alarms	26
Deep neural network	Zhang et al. (2018) [29]	Attacks against vehicles	False positive, Detection rate, Time per msg	4
Feed forward deep neural network	Kasongo et al. (2019) [30]	NSL-KDD dataset	Accuracy, Precision, Recall	0
Recurrent neural network	Kim et al. (2016) [31]	KDD Cup 1999 dataset	Detection Rate, FAR, Efficiency	91
Recurrent neural network	Taylor et al. (2016) [32]	Attacks against vehicles	ROC curve	81
Recurrent neural network	Loukas et al. (2017) [33]	Attacks against vehicles	Detection accuracy	26
Recurrent neural network	Yin et al. (2017) [34]	NSL-KDD dataset	Accuracy, TPR, FPR	109
Recurrent neural network	Tang et al. (2018) [35]	NSL-KDD dataset	Accuracy, Precision, Recall, F-measure	11
Recurrent neural network	Jiang et al. (2018) [36]	NSL-KDD dataset	Accuracy, Detection Rate, FAR	23
Recurrent neural network	Ferrag et al. (2019) [37]	CICIDS2017 dataset	Accuracy, Detection Rate, FAR	0
Convolutional neural network	Basumallik et al. (2019) [38]	IEEE-30 bus and IEEE-118 bus	Accuracy	1
Convolutional neural network	Fu et al. (2016) [39]	Credit card transaction data	Feature Score, Accuracy	49
Convolutional neural network	Zhang et al. (2018) [40]	Online transaction data	Accuracy, Precision, Recall	4
Convolutional neural network	Feng et al. (2019) [25]	KDD Cup 1999 dataset	Accuracy, Precision, Recall, F1-score	1
Convolutional neural network	Nasr et al. (2018) [41]	UMASS dataset	Accuracy, ROC Curve	5
Convolutional neural network	Zhang et al. (2019) [42]	CICIDS2017 dataset	Accuracy, Precision, Recall, F1âĹŠMeasure	0
Convolutional neural network	Zeng et al. (2019) [43]	ISCX 2012 IDS dataset	Precision, Recall, and F1 score	0
Convolutional autoencoder	Yu et al. (2017) [44]	Contagio-CTU-UNB dataset	Accuracy, Precision, Recall, F-measure, ROC curve	17
Restricted Boltzmann machine	Alrawashdeh et al. (2016) [45]	KDD Cup 1999 dataset	Accuracy	38
Restricted Boltzmann machine	Aldwairi et al. (2018) [46]	ISCX dataset	Accuracy, TPR, TNR	5
Restricted Boltzmann machine	Fiore et al. (2013) [47]	KDD Cup 1999 dataset	Accuracy, Speed, Comprehensibility, Time to learn	178
Restricted Boltzmann machine	Salama et al. (2011) [48]	NSL-KDD dataset	Accuracy	97
Restricted Boltzmann machine	Gao et al. (2014) [49]	KDD Cup 1999 dataset	Accuracy, Detection rate, FAR	72
Restricted Boltzmann machine	Alom et al. (2015) [50]	NSL-KDD dataset	Accuracy	61
Restricted Boltzmann machine	Yang et al. (2017) [51]	Real online network traffic	Precision, F1 score	19
Restricted Boltzmann machine	Otoum et al. (2019) [52]	KDD Cup 1999 dataset	Accuracy, Detection Rate, FNR, ROC curve, F1 score	9
Restricted Boltzmann machine	Karimipour et al. (2019) [53]	IEEE 39, 118, and 2848 bus systems	Accuracy, FPR, TPR	0
Deep belief network	Thamilarasu et al. (2019) [54]	IoT simulation dataset	Precision, Recall, F1-score	0
Deep belief network	Zhao et al. (2017) [55]	KDD Cup 1999 dataset	Detection, Detection rate, FAR	14
Deep belief network	Zhang et al. (2019) [56]	NSL-KDD dataset	Accuracy, Detection rate, FAR, Precision, Recall	1
Deep belief network	Aloqaily et al. [57]	NS-3 traffic and NSL-KDD dataset	Accuracy, Detection rate, FPR, FNR	36
Conditional deep belief networK	He et al. (2017) [58]	IEEE 118-bus and IEEE 300-bus	Accuracy, ROC curve	82
Deep auto-encoder	Shone et al. (2018) [59]	NSL-KDD dataset	Accuracy, Precision, Recall, False Alarm, F-score	73
Deep auto-encoder	Khan et al. (2019) [60]	UNSW-NB15 dataset	Accuracy, Precision, Recall, F-measure, FAR	0
Deep auto-encoder	Papamartzivanos et al. (2019) [61]	NSL-KDD dataset	Accuracy, FMeasure, Precision, Recall	3
Deep auto-encoder	Yang et al. (2019) [62]	NSL-KDD and UNSW-NB15	Accuracy, Precision, Detection rate, Recall, FPR, F1-score	0
Denoising auto-encoder	Abusitta et al. (2019) [63]	KDD Cup 1999 dataset	Accuracy, Test classification error	1
Stacked denoising auto-encoders	Wang et al. (2016) [64]	Heritrix dataset	Accuracy, Classification error, Precision, Recall, F-measure	40
Deep migration learning	Li et al. (2019) [65]	KDD Cup 1999 dataset	Detection rate, FAR, Precision, Missing rate	0
Self-Taught Learning	Javaid et al. (2016) [66]	KDD Cup 1999 dataset	Accuracy, Precision, Recall, F-measure	181
Replicator Neural Network	Cordero et al. (2016) [67]	MAWI dataset	Detecting anomalies, Detecting injected attacks	11

*No. of times cited (as of 22/06/2019)

classifier to handle huge network data. They used the NSL-KDD dataset, which contains 39 different attack types grouped into four attack classes. Their study shows that with 2 classes (i.e., normal and attack), the detection accuracy is high.

Kang et al. [23] proposed an intrusion detection system based on the deep neural network for vehicular networks. The attack scenario was performed on malicious data packets, which are injected into an in-vehicle controller area network

bus. The proposed system inputs the feature vector to the input nodes in order to classify packets into two classes (i.e., a normal packet and an attack packet). Based on the activation function, the outputs are computed (e.g., ReLU). Then, the next hidden layers are linked with these outputs. When the false positive error is less than 1-2%, the proposed system achieves a detection ratio of 99%.

To help classify cyber-attacks, Zhou et al. [24] proposed an intrusion detection system based on the deep neural network. Specifically, the system uses three phases, namely, 1) data acquisition (DAQ), 2) data pre-processing, and 3) deep neural network classification. The system achieves an accuracy of 0.963 for SVM model with learning rate 0.01, training epochs 10, and input units 86. The results show this approach to outperform slightly the following three machine learning approaches: 1) linear regression, 2) random forest, and 3) k-nearest neighborhood.

Feng et al. (2019) [25] describe a plug and play device that employs a capture tool to grab packets and deep learning detection model to detect Denial of Service (DoS) and privacy attacks in ad hoc networks. To detect XSS and SQL attacks, the proposed model uses two deep learning approaches, namely, convolutional neural network (CNN)and long short-term memory (LSTM). To detect DoS attacks, the proposed model uses a deep neural network. The study used the KDD CUP 99 dataset, which is split 30% for testing and 70% for training. In addition, the study reported accuracy of 0.57% and 0.78% for the detection of XSS attacks using the deep neural network and the convolutional neural network, respectively.

The study by Zhang et al. [26] is a good example of deep adversarial learning and statistical learning techniques to detect network intrusions. The study can identify a variety of network intrusions by exploiting data augmentation and advanced classification methods. The proposed system uses two components, including, the discriminator and the generator. The discriminator is used as an indicator to reject augmented data from real intrusion samples, while the generator is used to generate augmented intrusion data. To perform a deep neural network for ever-evolving network attacks, the work by Kim et al. [28] used the KDD 1999 data set. The proposed intrusion detection model uses two parameters, namely, four hidden layers and 100 hidden units. The ReLU function is used as the activation function and the stochastic optimization method for deep neural network training. The proposed model achieves an accuracy of approximately 99%.

Zhang et al. (2018) [29] introduced a intrusion detection system based on two-stage, named CAN IDS, for detecting malicious attacks against autonomous vehicles. A robust rule-based system is used in the first stage, while the second stage uses deep learning network for anomaly detection. Three datasets are used in the evaluation performance, including, Honda accord, Asia brand, and US brand vehicle. The training data contains only normal traffic from these three datasets, while the testing data contains normal traffic as well as malicious traffic under five types of attacks, including, drop attack, random attack, zero ID messages attack, replay attack, and spoofing attack.

B. Feed forward deep neural network

Feed forward deep neural network (FFDNN) was used for intrusion detection by Kasongo et al. [30]. They use an FFDNN with a filter-based feature selection approach in order to generate optimal subsets of features with minimum redundancy for wireless networks. The proposed intrusion detection system split the main training dataset between two main sets (i.e., the training dataset and the evaluation dataset). Then, it involves a two-way normalization process and a feature transformation process. Lastly, the proposed system uses the FFDNN for the models training and testing. The NSL-KDD dataset was used, and the KDDTrain+ and the KDDTest+ were chosen. With a learning rate of 0.05 and 30 neurons spread with 3 hidden layers, the performance evaluation show that the proposed system achieves an accuracy of 99.69%.

C. Recurrent neural network

The framework proposed by Kim et al. [31] use the KDD Cup 1999 dataset to perform long short term memory architecture to a recurrent neural model for intrusion detection. The study used (41 features) as an input vector (4 attacks and 1 non attack) as the output vector. They used a time step size 100, batch size 50, and epoch 500. The attack detection performance is reported as 98.8% among the total attack instances.

To detect cyber attacks against vehicles, Loukas et al. [33] proposed a cyber-physical intrusion detection system. The system uses both recurrent neural network architecture and deep multilayer perceptron, which achieves high accuracy with more consistency than standard machine learning techniques (e.g., k-means clustering and SVM). The system is evaluated under three types of attacks against a robotic vehicle, namely, command injection attack, denial of service attack, and malware attack targeting the network interface. Taylor et al. (2016) [32] proposed an anomaly detector scheme based on an artificial recurrent neural network architecture to detect attacks against vehicles. The Long Short-Term Memory (LSTM) is used as a recurrent neural network, which is trained to predict the new packet data values, and its errors are used as a signal to detect anomalies.

Yin et al. [34] attemtped to integrate a recurrent neural network in an IDS system for supervised classification learning. The study used the NSL-KDD dataset under three performance indicators, including accuracy, false positive rate, and true positive rate. The anomaly detection performance is reported as higher accuracy with the learning rate = 0.1 and hidden nodes = 80. The paper also states the benefits of a recurrent neural network for intrusion detection systems. In another study, Tang et al. [35] suggested a gated recurrent unit recurrent neural network for intrusion detection in software-defined networking. The paper states a detection rate of 89% using a minimum number of features. The NSL-KDD dataset is used in the network performance with four evaluation metrics, namely, recall, F-measure, precision, and accuracy. A multi-channel intrusion detection system that uses long short term memory recurrent neural networks is described by Jiang et al. [36]. The NSL-KDD dataset is used to evaluate the

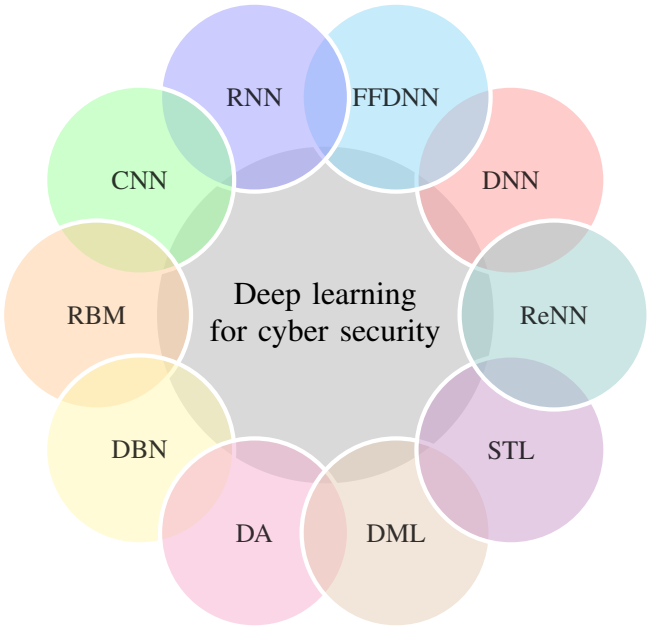


Fig. 1: Deep learning approaches used for cyber security intrusion detection. FFDNN: Feed forward deep neural network; CNN: Convolutional neural network; DNN: Deep neural network; RNN: Recurrent neural network; DBN: Deep belief network; RBM: Restricted Boltzmann machine; DA: Deep auto-encoder; DML: Deep migration learning; STL: Self-Taught Learning; ReNN: Replicator Neural Network.

performance of the proposed attack detection system. The performance of the long short term memory recurrent neural network is reported as 99.23% detection rate with a false alarm rate of 9.86% and an accuracy of 98.94%.

D. Convolutional neural network

Convolutional neural networks were used by Basumallik et al. [38] for packet-data anomaly detection in phasor measurement units-based state estimator. They use a convolutional neural network-based data filter in order to extract event signatures (features) from phasor measurement units. The IEEE-30 bus and IEEE-118 bus system are used as the phasor measurement unit buses. The study states a probability of 0.5 with 512 neurons at a fully connected layer and a 98.67% accuracy. The authors claim that convolutional neural network-based filter has a superior performance over other machine learning techniques, including RNN, LSTM, SVM, bagged, and boosted.

The framework developed by Fu et al. [39] uses a convolutional neural network in order to capture the intrinsic patterns of fraud behaviors, especially for credit card fraud detection. Zhang et al. [40] employed the convolutional neural network and used the commercial bank B2C online transaction data for training and testing. The data of one month were divided into training sets and test sets. The study states a precision rate of 91% and the recall rate of 94%. These results are increased by 26% and 2%, respectively, compared with the work proposed by Fu et al. [39].

In order to learn a correlation function, Nasr et al. [41] proposed an intrusion detection system, named DeepCorr, which is based on a convolutional neural network. Specifically, DeepCorr is based on two layers of convolution and three layers of a fully connected neural network. Experimentation showed that the best performance is with a learning rate of 0.0001, and for a false positive rate of 10^{-3}, DeepCorr achieves a true positive rate close to 0.8.

Based on two layers of the neural network, Zhang et al. [42] introduced an anomaly traffic detection model in which the first layer consists of the improved LetNet-5 convolutional neural network. The second layer uses long short-term memory. Specifically, the first layer is proposed to extract the spatial features, while the second layer is proposed to extract the temporal features of the flow. The performance on the CICIDS2017 dataset [68] was exceeded by 94%. Compared to other machine learning algorithms (e.g., NaiveBayes, Logistic Regression, Random Forest(RF), and Decision Tree), the proposed system can achieve high accuracy, precision, recall, and F1-measure. Therefore, the approach described by Zeng et al. [43] is a light-weight framework, named deep-full-range (DFR), for detection of novel attacks, encrypted traffic classification, and intrusion detection.

In the work by Yu et al. [44], a convolutional autoencoder was used to evaluate network intrusion on two intrusion detection datasets, namely, the CTU-UNB dataset and the Contagio-CTU-UNB dataset. The Theano tool is used to build the neural network model. The learning rates are 0.001 and 0.1 for the pretraining and fine-tuning process, respectively. The classification tasks include 6-class and 8-class classifications using the Contagio-CTU-UNB dataset. The ROC curve value of 6-class and 8-class classification is 0.99. In addition, the study achieves a 99.59% accuracy rate in the binary classification.

E. Restricted Boltzmann machine

The restricted Boltzmann machine was used for intrusion detection by Fiore et al. [47]. They use a discriminative restricted Boltzmann machine in order to combine the expressive power of generative models with good classification. The KDD Cup 1999 dataset was used with a set of 41 features and 97,278 instances. Salama et al. [48] combine the restricted Boltzmann machine and support vector machine for intrusion detection. The NSL-KDD dataset was used, whose training set contains a total of 22 training attack types, with an additional 17 types in the testing set. The study states that this combination shows a higher percentage of classification than when using support vector machine.

Alrawashdeh and Purdy [45] employed the restricted Boltzmann machine with a deep belief network and used the KDD 1999 data set, which contains 494,021 training records and 311029 testing record. The detection algorithm is implemented using C++ and Microsoft Visual Studio 2013. The study shows that the restricted Boltzmann machine classified 92% of the attacks. The paper compared the results to the work by Salama et al. [48], which shows both a higher accuracy and detection speed.

Aldwairi et al. [46] proposed a comparative study of restricted Boltzmann machines for cyber security intrusion detection. Specifically, the study demonstrates the performance of restricted Boltzmann machines to distinguish between normal and anomalous NetFlow traffic. The proposed study was applied to ISCX dataset [69], which the results show the highest accuracy of 78.7 ± 1.9% when the learning rate was set to 0.004. In addition, the true positive rate and true negative rate are high which 74.9 ± 4.6% and 82.4 ± 1.8%, respectively, at the learning rate 0.004.

Integrating multilayer unsupervised learning networks was attempted by Gao et al. [49] and applied in the intrusion recognition domain. The study uses a restricted Boltzmann machine, whose deep neural network training consists of two steps: 1) training the restricted Boltzmann machine of layers n, and 2) the parameters of the whole restricted Boltzmann machine are fine-tuned. The study shows that the performance on the KDD CUP 1999 dataset of deep belief network based on restricted Boltzmann machine is better than that of a support vector machine and an artificial neural network.

An intrusion detection system that uses a stack restricted Boltzmann machine is described by Alom et al. [50]. The main goal of this approach is to detect anomalous or malicious activities. The study uses the NSL-KDD dataset, and the attacks are classified into five categories. The results show that the proposed system achieves around 97.5% testing accuracy for only 40% of data used in training. Therefore, based on restricted Boltzmann machines, Yang et al. [51] proposed a new method using a support vector machine, named SVM-RBM, in order to provide improved traffic detection. The restricted Boltzmann machine is used for training features. During the process of feature extraction, the authors proposed to change the number of units in the hidden layers. Then, once the good features are obtained, the authors proposed to focus on training the model of a support vector machine in which the parameters in training follow the gradient descent algorithm. The proposed algorithm SVM-RBMS show the highest precision can reach 80%.

Otoum et al. [52] introduced a clustered intrusion detection system in wireless sensor networks, named RBC-IDS, which is based on the restricted Boltzmann machine. The RBC-IDS system uses the N clusters with C sensor nodes in each cluster. The study uses both the Network Simulator-3 (NS-3) and KDD Cup 1999 dataset in the performance evaluation. Compared to the adaptive machine learning-based IDS (ASCH-IDS) [70], the RBC-IDS system achieves the highest accuracy rate of 99.91% when the number of hidden layers is 3, while the ASCH-IDS archives 99.83%.

For securing the connectivity aspect of connected vehicles, Aloqaily et al. [57] proposed an intrusion detection system, named D2H-IDS, which is based on a deep belief network and decision tree. The deep belief network is used for data dimensionality reduction, while the decision tree is used for attacks classification. For data collection and processing, the D3H-IDS system adopts a cluster-head selection mechanism. The D2H-IDS system is evaluated through the NS-3 collected traffic along with the NSL-KDD dataset, which the results archives the highest detection rate compared to the work presented in [55]. For more information about the vehicular dataset, we refer the reader to the work presented in [71].

A deep unsupervised machine learning model is proposed by Karimipour et al. [53] for cyber security intrusion detection in large-scale smart grids. To build a computationally feature extraction, the symbolic dynamic filtering is used, which can discover causal interactions between the smart grids subsystems through dynamic Bayesian networks. To capture the patterns in system behavior, the authors proposed the use of a restricted Boltzmann machine. The results on IEEE 39 bus system under cyber-attack show that the proposed model can detect an attack with almost 99% accuracy and 98% true positive rate.

F. Deep belief network

The deep belief network was used for intrusion detection by Thamilarasu et al. [54]. They use a deep belief network to fabricate the feed-forward deep neural network for the Internet of Things. Specifically, the authors proposed a binary cross-entropy loss function in order to minimize the total cost in the IDS model. The Keras library, Cooja network simulator, and Texas Instruments sensor tags CC2650 are used on the performance evaluation. The Keras library is used for creating a sequential deep-learning model. The proposed model is tested against five attacksl: 1) sinkhole attack, 2) wormhole attack, 3) blackhole attack, 4) opportunistic service attack, and 5) DDoS attack. The results show a higher precision of 96% and a recall rate of 98.7% for detecting DDoS attacks.

In another study, Zhao et al. [55] suggested an IDS framework using deep belief network and probabilistic neural network. The study uses the KDD CUP 1999 dataset to evaluate the intrusion detection model with 10% training dataset and the 10% testing dataset. The experiment result shows that the method performs better than three models, namely, 1) the traditional probabilistic neural network, 2) principal component analysis with the traditional probabilistic neural network and 3) unoptimized deep belief network with probabilistic neural network.

The study by Zhang et al. [56] is a good example of the combination of improved genetic algorithm and deep belief network for cyber security intrusion detection. The study uses multiple restricted Boltzmann machines, which are mainly executing unsupervised learning of pre-processed data. The deep belief network module is divided into two steps in the training phase: 1) each restricted Boltzmann machine is trained separately, and 2) the last layer of the deep belief network is set as the back propagation neural network. The performance evaluation using NSL-KDD dataset shows a detection rate of 99%.

To detect false data injection attack in the supervisory control and data acquisition system, He et al. [58] proposed an intrusion detection system based on the extended deep belief network architecture. The study exploits conditional Gaussian-Bernoulli restricted Boltzmann machine in order to extract high-dimensional temporal features. The proposed system can reduce the complexity of training and execution time of the deep learning architecture compared to work proposed by

Taylor et al. [72]. The performance evaluation on the IEEE 118-bus power test system and the IEEE 300-bus system show the highest accuracy of detection at 98.5%.

G. Deep auto-encoder

The deep auto-encoder was used by Shone et al. (2018) [59] for cyber security intrusion detection. They use an auto-encoder featuring non-symmetrical multiple hidden layers to facilitate improved classification results compared with deep belief networks. The study uses the KDD Cup '99 and NSL-KDD datasets with five metrics performances, including, accuracy, precision, recall, false alarm, and F-score. The results on the KDD Cup '99 dataset evaluation show that the proposed model is able to offer an average accuracy of 97.85%, which is better compared to the work in [45]. In addition, the results on the NSL-KDD dataset evaluation show that the proposed model offered a total accuracy rate of 85.42%, which is an improvement upon the deep belief network model by 5%.

Khan et al. [60] proposed an intrusion detection system based on the two-stage deep learning model, named TSDL. The TSDL model uses a stacked auto-encoder with a soft-max classifier, which is composed of three main layers, namely, 1) the input layer, 2) the hidden layers, and 3) the output layer. These three layers employ a feed-forward neural network similar to a multi-layer perceptron. The study uses two public datasets, including, KDD99 and UNSW-NB15 datasets. The results on KDD99 dataset achieve high recognition rates, up to 99.996%. In addition, the results on UNSW-NB15 dataset achieve high recognition rates, up to 89.134%.

To design a self-adaptive and autonomous misuse intrusion detection system, Papamartzivanos et al. [61] propose the use of auto-encoder techniques. Specifically, the proposed system is based in four phases, including 1) Monitor, 2) Analyze, 3) Plan, 4) Execute, and 5) Knowledge. The monitor phase determines any alteration event that requires an intrusion detection system adaptation. The analyze phase uses network audit tools (e.g., Argus and CICFlowMeter) to perform the transformation of the raw network traffic into network flows. The plan phase uses a sparse autoencoder to learn representations of the input data. The execute phase is accountable for storing purposes. However, the study uses two datasets in performance evaluation, including KDDCup'99 and NSL-KDD. The results show that the average accuracy of the static model is 59.71% while for the adaptive model it is 77.99%

The combination of an improved conditional variational autoencoder and deep neural network was used by Yang et al. [62] for cyber security intrusion detection. The proposed study consists of three phases: 1) training, 2) generating new attacks and 3) detecting attacks. The training phase consists of optimizing the loss of the encoder and the decoder. The phase of generating new attacks uses a multivariate Gaussian distribution as the distribution. The phase of detecting attacks employs a deep neural network to detect attacks. To validate the proposed model, the NSL-KDD and UNSW-NB15 datasets are used, which the default learning rate of the Adam optimizer is 0.001. The results show the highest accuracy of 89.08% and detection rate of 95.68% on the UNSW-NB15 dataset.

To construct a deep neural network, Abusitta et al. [63] uses a denoising autoencoder as a building block for cyber security intrusion detection. The denoising autoencoder is used to learning how to reconstruct intrusion detection systems feedback from partial feedback. The KDD Cup 99 dataset is used on the performance evaluation, which the results show that the proposed model can achieve detection accuracy up to 95%. Therefore, stacked denoising auto-encoders is used by Wang et al. (2016) [64] for detecting malicious JavaScript code. The study uses a dataset, which is composed of 12 320 benign and 14 783 malicious JavaScript samples. With three layers of auto-encoders and 250 hidden units, the experimental results show an optimal choice for building an intrusion detection system based on the deep learning technique.

H. Deep migration learning

Deep migration learning is used by Li et al. [65] for cyber security intrusion detection. Specifically, the study combines deep learning model with the intrusion detection system. According to this study, deep migration learning can be divided into four categories, including parameter migration technique, sample migration technique, related knowledge migration technique, and feature representation migration technique. The study uses the KDD CUP 99 dataset as input for experimental data with 10% of the training set as experimental data. During the experiment, the study selects randomly sampled 10,000 datasets as training sets as well as sampled 10,000 data sets as experimental test sets. The results show a detection rate of 91.05% and a false alarm rate of 0.56%.

I. Self-Taught Learning

Self-taught learning is used by Javed et al. [66] for cyber security intrusion detection. The proposed technique uses phases for the classification, including learning feature representation and learned representation is used for the classification task. The study uses the NSL-KDD dataset on the performance evaluation. The proposed system is applied in three different types of classification: 1) 2-class (i.e., normal and anomaly), 2) 5-class (i.e., normal and four different attack categories), and 3) 23-class (i.e., normal and 22 different attacks). To evaluate the classification accuracy of self-taught learning for these three types of classification, the study applied 10-fold cross-validation on the training data. The results show an f-measure value of 75.76%.

J. Replicator Neural Network

The replicator neural networks are used by Cordero et al. [67] for cyber security intrusion detection. The study uses the dropout technique to find anomalies. The entropy extraction is comprised of three steps. The first step is the aggregation of packets. The second step is the segmentation of the flows into time windows. The last step is the selection of features of interest from the flows. The MAWI dataset is used on the performance evaluation, in which the injected synthetic attacks (e.g., SYN DDoS) are integrated into the dataset.

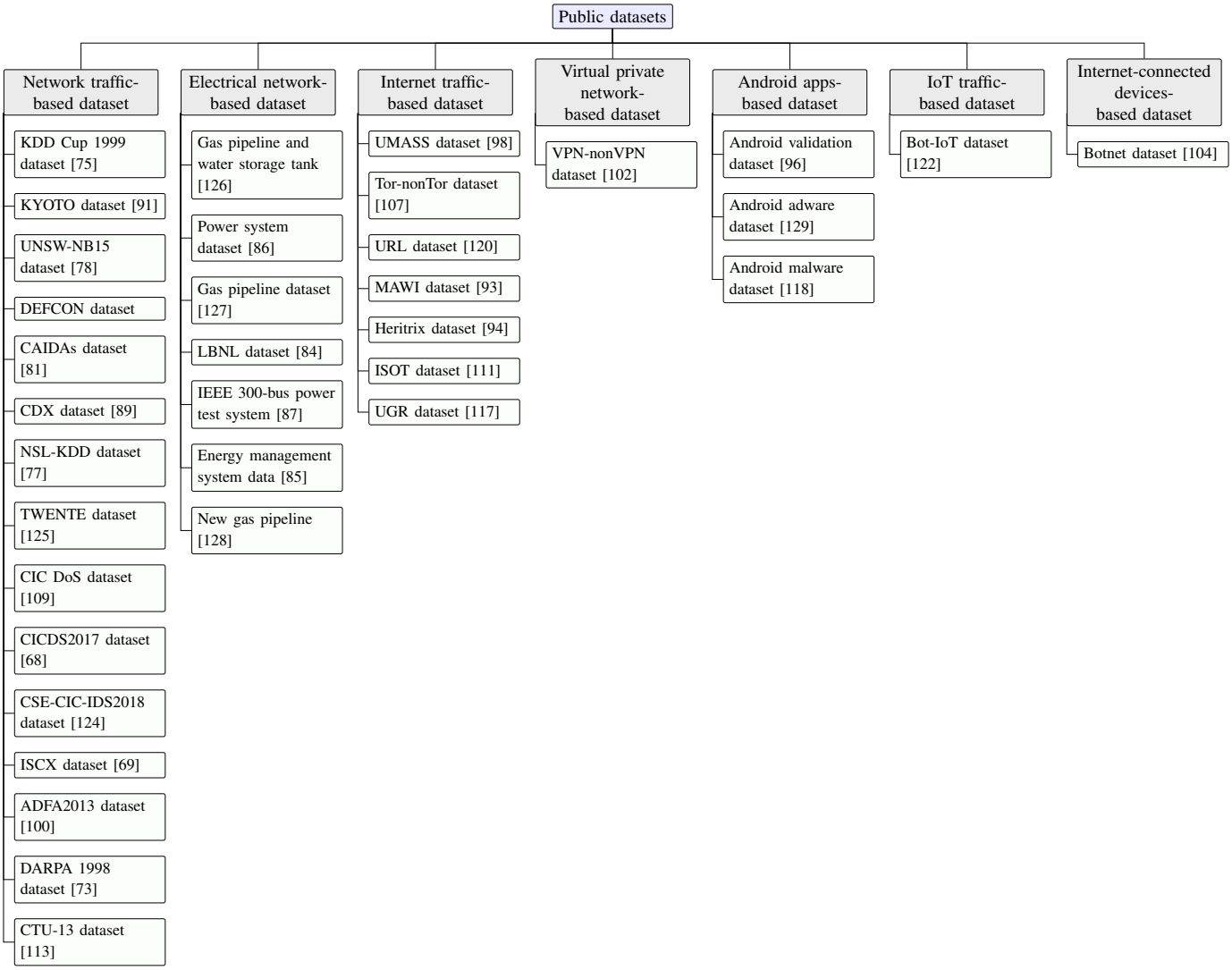


Fig. 2: Classification of public datasets for cyber security intrusion detection.

IV. Public datasets

Table II lists the representative deep learning approaches papers applied to intrusion detection that were reviewed, including the number of times they have been cited and the dataset used. We can observe that most papers use four datasets, including the UNSW-NB15 dataset, the KDD Cup 1999 dataset, and the NSL-KDD dataset. However, there are other datasets that can be used for cyber security intrusion detection. We present these datasets in Table III. Based on the content of each dataset, we classify them into the following seven main categories: 1) network traffic-based dataset, 2) electrical network-based dataset, 3) internet traffic-based dataset, 4) virtual private network-based dataset, 5) android apps-based dataset, 6) IoT traffic-based dataset, and 7) internet-connected devices-based dataset, as presented in Figure 2.

A. Network traffic-based dataset

1) DARPA 1998 dataset: [73] This dataset is based on the network traffic and audit logs, and was first made available in February 1998. The training data contains seven weeks of network-based attacks, while the testing data contains two weeks of network-based attacks. According to work Sharafaldin et al. [130], this dataset does not represent real-world network traffic.

2) KDD Cup 1999 dataset: [75] This dataset is based on DARPA'98 IDS evaluation program and contains seven weeks of network traffic, which consists of approximately 4,900,000 vectors. The simulated attacks are categorized into the following four groups: 1) User to Root attack (U2R), 2) Remote to Local attack (R2L), 3) Probing attack, and 4) Denial of Service attack (DoS). The KDD Cup 1999 dataset contains 41 features, which are categorized into the following three classes: 1) basic features, 2) traffic features, and 3) content features. The basic features are extracted from a TCP/IP connection. The traffic features are divided into two groups (i.e., "same host" features, "same service" features). The content features concerns suspicious behavior in the data portion. Note that this dataset is the most widely used dataset for the evaluation of intrusion detection models.

TABLE III: Public datasets for cyber security intrusion detection

Public Dataset	Year	Publicly available	Details	Cited*
DARPA dataset	1998	[73]	[74]	1069
KDD Cup 1999 dataset	1999	[75]	[76]	N/A
NSL-KDD dataset	2009	[77]	[76]	1630
UNSW-NB15 dataset	2015	[78]	[79]	202
DEFCON dataset	2000	N/A	[80]	12
CAIDAs dataset	2017	[81]	[82]	18
LBNL dataset	2016	[83]	[84]	7
ICS cyber attack dataset	2015	[85]	[86]	124
IEEE 300-bus power test system	N/A	N/A	[87]	171
CDX dataset	2013	[88]	[89]	8
KYOTO dataset	2006	[90]	[91]	12
MAWI dataset	2011	[92]	[93]	182
Heritrix dataset	2010	[94]	[95]	N/A
TWENTE dataset	2014	[96]	[97]	222
UMASS dataset	2018	[98]	[41]	5
ISCX dataset	2012	[69]	[99]	453
ADFA2013 dataset	2013	[100]	[101]	147
VPN-nonVPN dataset	2016	[102]	[103]	49
Botnet dataset	2014	[104]	[105]	99
Android validation dataset	2014	[96]	[106]	33
Tor-nonTor dataset	2017	[107]	[108]	34
CIC DoS dataset	2017	[109]	[110]	18
ISOT dataset	2008	N/A	[111]	98
CTU-13 dataset	2013	[112]	[113]	244
SSHCure dataset	2014	[114]	[115]	37
UGR dataset	2016	[116]	[117]	12
Android malware dataset	2018	[118]	[119]	1
URL dataset	2016	[120]	[121]	7
CICDS2017 dataset	2017	[68]	[6]	87
Bot-IoT dataset	2018	[122]	[123]	2
CSE-CIC-IDS2018 dataset	2018	[124]	N/A	N/A

*No. of times cited (as of 22/06/2019)

3) NSL-KDD dataset: [77] This dataset is proposed by Tavallaee et al. [76] and is recommended to solve some of the inherent problems of the KDD'99 dataset. Compared to the original KDD dataset, the NSL-KDD dataset has the following improvements: 1) it does not include redundant records, 2) it does not include duplicate records, 3) the number of selected records is organized as the percentage of records (e.g., KDDTrain+_20Percent.ARFF), and 4) the number of records is reasonable. Note that many papers on intrusion detection use both datasets together in performance evaluation (i.e., KDD Cup 1999 dataset and NSL-KDD dataset), and they typically find that the best results are found in the NSL-KDD dataset.

4) UNSW-NB15 dataset: [78] This dataset is created by four tools, namely, IXIA PerfectStorm tool, Tcpdump tool, Argus tool, and Bro-IDS tool. These tools are used to create some types of attacks, including DoS, Exploits, Generic, Reconnaissance, Shellcode, and Worms. The UNSW-NB15 dataset contains approximately two million and 540,044 vectors with 49 features. In addition, Moustafa et al. [131] published a partition from this dataset which contains the training set (175,341 vectors) and the testing set (82,332 vectors).

5) DEFCON dataset: This dataset is generated with two versions, including, DEFCON-8 (2000) and DEFCON-10 (2002). Attacks in DEFCON-8 dataset contains ports scan and buffer overflow. Attacks in DEFCON-10 dataset contains probing and non-probing attacks (e.g., bad packet, ports scan, port sweeps, etc.). Both versions are used by Nehinbe Ojo Joshua [80] for reclassifying network intrusions.

6) CAIDAs dataset: [81] This dataset is proposed by the Center of Applied Internet Data Analysis, which contains different datasets, including, CAIDA DDOS, CAIDA Internet traces 2016, and RSDoS Attack Metadata (2018-09). Specifically, the CAIDA DDOS includes one-hour DDoS attack traffic split of 5-minute pcap files that are passive traffic traces from CAIDA's Equinix-Chicago. The RSDoS Attack Metadata (2018-09) includes the randomly spoofed denial-of-service attacks inferred from the backscatter packets collected by the UCSD Network Telescope.

7) CDX dataset: This dataset is created by Homoliak et al. [89] during network warfare competition, which contains malicious and legitimate TCP communications on network services. These services are vulnerable to buffer overflow attacks. However, there are four types of CDX 2009 vulnerable servers, including, Postfix Email FreeBSD, Apache Web Server Fedora 10, OpenFire Chat FreeBSD, and BIND DNS FreeBSD.

8) KYOTO dataset: [91] This dataset is based on real three year-traffic data, which is created using four tools, including, honeypots, darknet sensors, e-mail server and web crawler. The Kyoto dataset contains 24 statistical features, which 14 features were extracted based on KDD Cup 99 data set and 10 additional features.

9) TWENTE dataset: [125] This dataset is collected over a period of 6 days, which is resulted in 14.2M flows and 7.6M alerts. TWENTE dataset presents a subdivision using three IP protocols, including, UDP, TCP, and ICMP.

10) CIC DoS dataset: [109] This dataset contains application layer DoS attacks with 4 types of attacks using different tools. The CIC DoS dataset is proposed by Jazi et al. [110], which application layer DoS attacks are generally seen in high-volume (e.g., high-volume HTTP attacks generated using HULK (HTTP Unbearable Load King)) or low-volume variations (e.g., Low-volume DoS attacks). The High-volume HTTP attacks include DoS improved GET (Goldeneye), DDoS GET(ddossim), and DoS GET (hulk). The Low-volume HTTP attacks include slow-send body (Slowhttptest), slow send body (RUDY), slow-send headers (Slowhttptest), slow send headers (Slowloris), and slow-read (Slowhttptest).

11) CICDS2017 dataset: [68] This dataset contains data captured from Monday, July 3, 2017, to Friday, July 7, 2017. The CICIDS2017 dataset is proposed by Sharafaldin et al. [6], which implements attacks include Brute Force SSH, DoS, Heartbleed, Web Attack, Infiltration, Botnet and DDoS, and Brute Force FTP. The CICFlowMeter tool is used to extract 80 network flow features from the generated network traffic. The CICFlowMeter tool [132] is used to extract 80 network flow features from the generated network traffic. In addition, the CICIDS2017 dataset extracts the abstract behavior of 25 users based on some protocols such as FTP, HTTPS.

12) CSE-CIC-IDS2018 dataset: [124] This dataset is proposed by the Communications Security Establishment (CSE) & the Canadian Institute for Cybersecurity (CIC). The dataset CSE-CIC-IDS2018 dataset includes seven different attack scenarios, including, Heartbleed, Brute-force, DoS, DDoS, Web attacks, Botnet, and infiltration. Similarly to CICDS2017 dataset [68], the CICFlowMeter tool [132] is used to extract 80 network flow features from the generated network traffic.

13) ISCX dataset: [69] This dataset is created by Shiravi et al. [99] , which consists of the 7 days of network activity (normal and malicious). The network activity malicious includes 1) Infiltrating the network from inside, 2) HTTP Denial of Service, 3) Distributed Denial of Service, and 4) Brute Force SSH. However, there are two general classes of profiles used in the ISCX dataset, namely, 1) profiles attempt to describe an attack scenario in an unambiguous manner and 2) profiles encapsulate extracted mathematical distributions or behaviors of certain entities.

14) ADFA2013 dataset: [100] This dataset is proposed by Creech and Hu [101], [133] which uses payloads and vectors to attack the Ubuntu OS. The payloads include password brute-force, add new superuser, java based meterpreter, linux meterpreter payload, and C100 Webshell. The dataset structure contains three data types, namely, 1) normal training data, 2) normal validation data, and 3) attack Data. The normal training data contains 4373 traces. The normal validation data contains 833 traces. The attack data contains 10 attacks per vector.

B. Electrical network-based dataset

1) LBNL dataset: [84] This dataset is collected using the uPMU at the Lawrence Berkeley National Laboratory electrical network. The uPMU is micro-phasor measurement units, which produces 12 streams of 120 Hz high precision values with timestamps accurate to 100 ns. This dataset can be used for microgrid synchronization as well as characterization of loads and distributed generation.

2) ICS cyber attack dataset: [85] This dataset contains five different datasets, including, 1) Power System Datasets, 2) Gas Pipeline Datasets, 3) Energy Management System Data, 4) New Gas Pipeline, and 5) Gas Pipeline and Water Storage Tank. The Power System dataset contains 37 scenarios, which are divided into 8 natural events, 1 no events, and 28 attack events. There are three categories of attacks, including, 1) relay setting change, 2) remote tripping command injection, and 3) data injection. These datasets can be used for cyber security intrusion detection in the industrial control systems [86], [134], [127], [135], [128].

3) IEEE 300-bus power test system: This dataset provides a topological and electrical structure of power grid, which is used especially for the detection of false data injection attacks in the smart grid. The system has 411 branches, and average degree (< k >) of 2.74. For more details about this standard test system, we refer the reader to the work presented by Hines et al. [87]. The IEEE 300-bus power test system has been used for multiple works related to cyber-attack classification [38], [53], [58].

C. Internet traffic-based dataset

1) UMASS dataset: [98] This dataset contains two different datasets, including, 1) strong flow correlation attacks and 2) simple timing attack on OneSwarm. The strong flow correlation attacks are proposed by Nasr et al. [41], which they used several Tor clients to browse the top 50,000 Alexa websites over Tor. The simple timing attack on OneSwarm is proposed by Bissias et al. [136], which the attacks adhere to the restrictions of a constrained generally applicable criminal procedure. Specifically, there are three independent attacks, including, an attack based on timing information, an attack based on query forwarding, and an attack based on TCP throughput.

2) Tor-nonTor dataset: [107] This dataset is proposed by Lashkari et al. [108], which contains 8 types of traffic (VOIP, chat, audio-streaming, video-streaming, mail, P2P, browsing, and File Transfer) from more than 18 representative applications (e.g., Spotify, skype, facebook, gmail, etc.). For the non-Tor traffic, this dataset used benign traffic from VPN project created in [103].

3) URL dataset: [120] This dataset is proposed by Mamun et al. [121], which contains five different types of URLs, including, 1) Benign URLs, 2) Spam URLs, 3) Phishing URLs, 4) Malware URLs, and 5) Defacement URLs. The Benign URLs contains 35,300 benign URLs, which they are collected from Alexa top websites. The Spam URLs contains 12,000 spam URLs, which they are collected from the WEBSPAM-UK2007 dataset. The Phishing URLs contains 10,000 phishing URLs, which they are collected from a repository of active phishing sites, named OpenPhish. The Malware URLs contains 11,500 URLs, which they are collected from a maintained list of malware sites, named DNS-BH. The Defacement URLs contains 45,450 URLs, which they are collected from Alexa ranked trusted websites hosting fraudulent or hidden URL.

4) MAWI dataset: [92] This dataset contains daily traces of traffic in the form of packet captures, which is captured from a trans-Pacific link between Japan and the United States. The MAWI dataset can be used to study anomaly detectors, internet traffic characteristics, and traffic classifiers. For example, Cordero et al. [67] used MAWI dataset with injected synthetic attacks in order to study to anomaly-based intrusion detection using a replicator neural network. For more details about the MAWI dataset, we refer the reader to the work presented by Fontugne et al. [93].

D. Virtual private network-based dataset

1) VPN-nonVPN dataset: [102] This dataset is proposed by Draper-Gil [103], which s captured a regular session and a session over virtual private network (VPN). Specifically, the VPN-nonVPN dataset consists of labeled network traffic, including, Web Browsing (e.g., Firefox), Email (e.g., SMPTS), Chat (e.g.,Skype), Streaming (e.g.,Youtube), File Transfer (e.g., SFTP), VoIP (e.g., Hangouts voice calls), and P2P (uTorrent).

E. Android apps-based dataset

1) Android validation dataset: [96] This dataset consists of 72 original apps with the following operations: replace icons,

replace files, insert junk code, different aligns, insert junk files, and replace strings. The android validation dataset is proposed by Gonzalez et al. [106] for finding different relationships among the Android apps. They extracted two features, namely, 1) Meta-information (accompanies each .apk file) and 2) N-grams (characterizing the .dex file). In addition, the android validation dataset introduced the following definitions that outline relations between apps: Twins, Siblings, False siblings, Stepsiblings, False stepsiblings, and Cousins. The original base set contains 72 apps, while the complete set with transformed apps contains 792 apps.

2) Android adware dataset: [129] This dataset is generated from 1900 applications with three categories, including, Adware (250 apps), General Malware (150 apps), and Benign (1500 apps). The details of the Android adware dataset is discussed by Lashkari et al. [137]. The category adware consisting the following popular families: Airpush, Dowgin, Kemoge, Mobidash, and Shuanet. To investigate the relationships between each app's category (adware, general malware, and benign), the authors used a lightweight detector of Android apps similarity, named Droidkin [106]. The Android smart phones (NEXUS 5) is used for running apps and a gateway is used for capturing the generated traffic, which the traffics are labeled in three categories (adware, benign, and general malware).

3) Android malware dataset: [118] This dataset is named CICAndMal2017, which contains both malware and benign applications. The CICAndMal2017 dataset is proposed by Shiravi et al. [99] from Googleplay market published in 2015, 2016, 2017. The malware samples in the CICAndMal2017 dataset are classified into the following four categories: 1) Adware (e.g., Dowgin, Ewind, Selfmite, Shuanet, ...etc), 2) Ransomware (e.g., Charge, Jisut, LockerPin, Pletor, WannaLocker, ...etc), 3) Scareware (e.g., AndroidDefender, AndroidSpy, VirusShield, Penetho, ...etc), and 4) SMS Malware (e.g., BeanBot, FakeInst, Mazarbot, Zsone, ...etc). In addition, the CICAndMal2017 dataset contains network traffic features (.pcap files) with more than 80 features.

F. IoT traffic-based dataset

1) Bot-IoT dataset: [122] This dataset contains more than 72.000.000 records, which includes DDoS, DoS, OS and Service Scan, Keylogging and Data exfiltration attacks. The Bot-IoT dataset is proposed by Koroniotis et al. [123], which is new for the IoT environment compared to previous datasets. The authors employed the Node-red tool to simulate the network behavior of IoT devices. To link machine-to-machine (M2M) communications, the dataset uses the MQTT protocol, which is a lightweight communication protocol. However, there are five IoT scenarios used in the testbed, namely, weather station, smart fridge, motion activated lights, remotely activated garage door, and smart thermostat.

G. Internet-connected devices-based dataset

1) Botnet dataset: [104] This dataset is proposed by Beigi et al. [105], which is divided into training and test datasets that included 7 and 16 types of botnets, respectively. The distribution of botnet types in the training dataset includes Neris, Rbot, Virut , NSIS, SMTP Spam, Zeus, and Zeus control (C&C). The distribution of botnet types in the test dataset includes, Neris, Rbot, Menti, Sogou...etc. The botnet topologies can be centralized, distributed (e.g., P2P) or randomized. The features used are categorized into four groups, namely, Byte-based, Packet-based, Time, and Behavior-based.

In our comparative study, we use two new real traffic datasets, namely, the CSE-CIC-IDS2018 dataset and the Bot-IoT dataset.

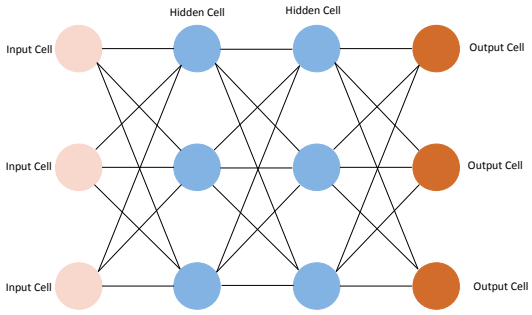


Fig. 3: Deep neural network.

V. Deep Learning Approaches

According to Deng and Yu [138], deep learning approaches can be classified into two models, namely, deep discriminative models and generative/unsupervised models. The deep discriminative models include three approaches, namely, recurrent neural networks, deep neural networks, convolutional neural networks. The generative/unsupervised models include four approaches, namely, deep autoencoders, restricted Boltzmann machine, and deep Boltzmann machines, and deep belief networks. Depending on how these deep learning approaches are intended for use, these techniques can be classified into three categories as follows; Category 1: Deep approaches for supervised learning, Category 2: Deep approaches for unsupervised or generative learning and Category 3: Hybrid deep approaches.

A. Deep discriminative models

1) Deep neural networks (DNNs): Deep Neural Network is multilayer perceptrons (MLP) with a number of layers superior to three. MLP is a class of feed forward artificial neural network, which is defined by the n layers that compose it and succeed each other, as presented in Figure 3.

The layer $M \in [1, N]$ of a DNN network is defined by $D_M(a_M, \alpha_M, n_M)$. $a_M \in \mathrm{N}$ is the number of neurons in the layer. $\alpha_M : \mathbb{R}^{a_{M-1}} \rightarrow \mathbb{R}^{a_M}$ is the affine transformation defined by the matrix W_M and the vector b_M. $n_M : \mathbb{R}^{a_M} \rightarrow \mathbb{R}^{a_M}$ is the transfer function of the layer M. The matrix W_M is called the weight matrix between the layer $M-1$ and the layer M. The vector b_M is called the bias vector of the layer M. Refer to Figure 3 and [139], deep neural network algorithm based on MLP is described as Algorithm 1.

Algorithm 1 DNN network based on MLP

Choose a learning pair $(x,\ c)$;
$h_0 = x$;
for $M = 1$ to N **do**
    $g_M = n_M\left(h_{M-1}\right) = W_M \times h_{M-1} + b_M$;
    $h_M = \alpha_M(g_M)$
end for

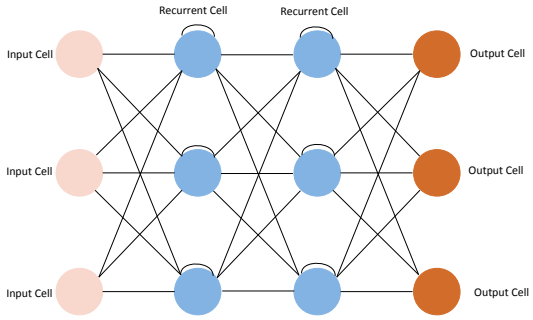


Fig. 4: Recurrent neural network

2) Recurrent neural networks (RNNs): A recurrent neural network is a neuron network, which the connection graph contains at least one cycle. There are many types of RNNs such as Elman networks proposed by [140], Jordan networks proposed by [141] and Echo State networks proposed by [142]. Currently, RNN based on Long Short-Term Memory (LSTM) is the most used. The RNN is defined by adding an interconnection matrix $VW_M \in \mathbb{R}^{a_M \times a_M}$ to the layer $M \in [1, N]$ in order to obtain a layer M' of the recurrent network. Refer to Figure 4 and [143], recurrent neural network algorithm is described as Algorithm 2.

3) Convolutional neural networks (CNNs): A convolutional neural network is defined as a neural network that extracts features at a higher resolution, and then convert them into more complex features at a coarser resolution, as presented in Figure 5. There are many types of CNNs such as ZFNet proposed by [144], GoogleNet proposed by [145], and ResNet proposed by [146]. Therefore, CNN is based on three types of layers, including, convolutional, pooling, and fully-connected layers. Refer to [147], the feature value at location (x, y) in the k-th feature map of M-th layer can be calculated as follows:

$$feature^M_{x,y,k} = W^{M^T}_k X^M_{x,y} + b^M_k \tag{1}$$

where $X^M_{x,y}$ is the input patch centered at location (x, y), W^M_k is the weight vector of the k-th filter, and b^M_k is bias

Algorithm 2 Recurrent neural network

Choose a learning pair $(x(t),\ c(t))$;
$h_0(t) = x(t),\ \forall t \in [1, t_f]$;
for $M = 1$ to N **do**
    for $t = 1$ to t_f **do**
        $g_M(t) = W_M \times h_{M-1}(t) + VW_M \times h_M(t-1) + b_M$;
        $h_M(t) = \alpha_M(g_M(t))$;
    end for
end for

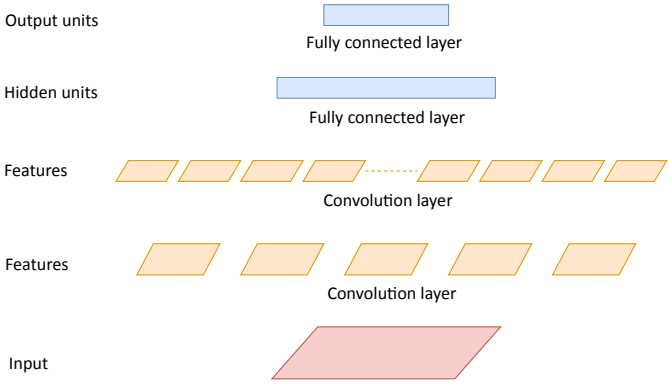


Fig. 5: Convolutional neural network

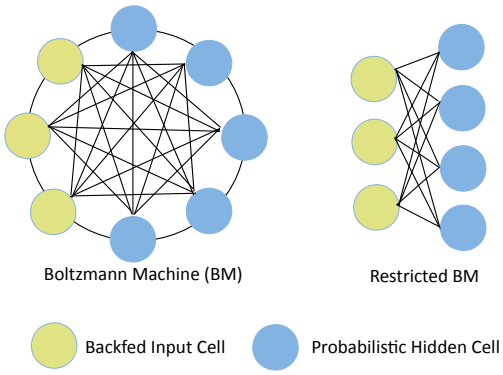


Fig. 6: Restricted Boltzmann machine

term of the M-th layer.

The activation value $activ^M_{x,y,k}$ and pooling value $pool^M_{x,y,k}$ of convolution feature $feature^M_{x,y,k}$ can be calculated as follow

$$activ^M_{x,y,k} = activation(feature^M_{x,y,k}) \tag{2}$$

$$pool^M_{x,y,k} = pooling\left(feature^M_{a,c,k}\right),\ \forall (a,c) \in \mathcal{R}_{x,y} \tag{3}$$

where $\mathcal{R}_{x,y}$ is a local neighbourhood around location at location (x, y). The nonlinear activation function $activation(\cdot)$ are be ReLU, sigmoid, and tanh. The pooling operation $pooling(\cdot)$ are average pooling and max pooling.

B. Generative/unsupervised models

1) Restricted Boltzmann machine (RBMs): An RBM is an undirected graphic model $G = \{W_{ij}, b_i, c_j\}$, as presented in Figure 6. There are two layers, including, the hidden layer and the visible layer. The two layers are fully connected through a set of weights W_{ij} and $\{b_i, c_j\}$. Note that there is no connection between the units of the same layer. Refer to [148], the configuration of the connections between the visible units and the hidden units has an energy function, which can be defined as follow:

$$En(V, H, G) = -\sum_i \sum_j V_j H_j W_{ij} - \sum_{i \in V} b_i V_i - \sum_{j \in H} c_j H_j \tag{4}$$

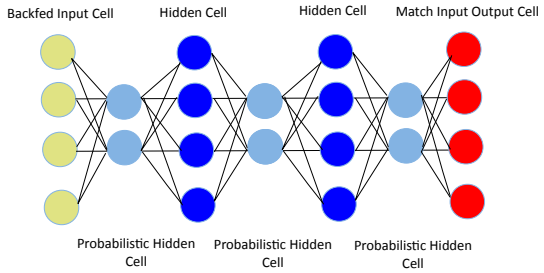


Fig. 7: Deep belief network.

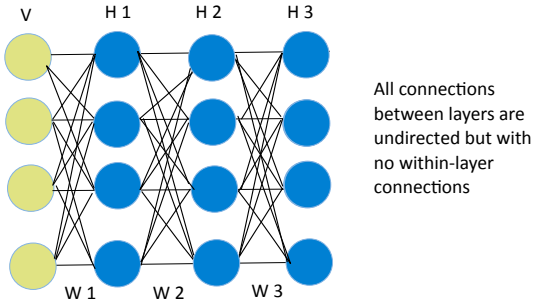


Fig. 8: Deep Boltzmann machine.

Based on this energy function, the probability of each joint configuration can be calculated according to the Gibbs distribution as follow:

$$Prob\,(V,H,G) = -\frac{1}{Z(G)}e^{-En(V,H,G)} \tag{5}$$

where Z is the partition function, which can be calculated as follow:

$$Z\,(G) = \sum_{V\in\mathcal{V}}\sum_{H\in\mathcal{V}} e^{-En(V,H,G)} \tag{6}$$

where curved letters $\mathcal{V}$ and $\mathcal{V}$ are used to denote the space of the visible and hidden units, respectively.

2) Deep belief networks (DBNs): A DBN is multi-layer belief network where each layer is Restricted Boltzmann Machine, as presented in Figure 7. The DBN contains a layer of visible units and a layer of hidden units. The layer of visible units represent the data. The layer of hidden units learns to represent features. Refer to [149], the probability of generating a visible vector, V , can be calculated as:

$$Prob(V) = \sum_{H} Prob\,(H \mid W)\,Prob(V|H,W) \tag{7}$$

where $Prob\,(H \mid W)$ is the prior distribution over hidden vectors.

3) Deep Boltzmann machines (DBMs): A DBM is a network of symmetrically coupled stochastic binary units, which contains a set of visible units and a sequence of layers of hidden units, as presented in Figure 8. Refer to [150], a DBM with three hidden layers can be defined by the energy of the state $\{V,H\}$ as:

$$En\,(V,H,G) == -V^T W^1 H^1 - V^1 W^2 H^2 - V^2 W^3 H^3 \tag{8}$$

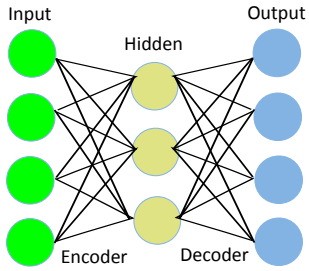


Fig. 9: Deep auto encoder.

TABLE IV: Attack Types in CSE-CIC-IDS2018 dataset

Category	Attack Type	Flow Count	Training	Test
Brute-force	SSH-Bruteforce	230	184	46
	FTP-BruteForce	611	489	122
Web attack	Brute Force -XSS	187589	7504	1876
	Brute Force -Web	193360	15469	3867
	SQL Injection	87	70	17
DoS attack	DoS attacks-Hulk	466664	18667	4667
	DoS attacks-SlowHTTPTest	139890	55956	13989
	DoS attacks-Slowloris	10990	4396	1099
	DoS attacks-GoldenEye	41508	16603	4151
DDoS attack	DDOS attack-HOIC	686012	27441	6860
	DDOS attack-LOIC-UDP	1730	1384	346
	DDOS attack-LOIC-HTTP	576191	23048	5762
Botnet	Bot	286191	11448	2862
Infilteration	Infilteration	161934	6478	1620
Benign	/	12697719	50791	12698
Total	/	15450706	231127	57782

where $H = \{H^1, H^2, H^3\}$ are the set of hidden units, and $G = \{W^1, W^2, W^3\}$ are the model parameters. The probability that the model assigns to a visible vector V can be defined as:

$$Prob\,(V,G) = \frac{1}{Z(G)}\sum_{H} e^{-En(V,H,G)} \tag{9}$$

4) Deep auto encoders (DA): An autoencoder is composed of both the encoder and the decoder, as presented in Figure 9. Refer to [151], these two parts can be defined as follow:

$$encoder_G\,(x) = s(Wx + b) \tag{10}$$

$$decoder_{G'}\,(y) = s\,(W'y + b') \tag{11}$$

where $G = \{W, b\}$; $G' = \{W', b'\}$; W is a $d' \times d$ weight matrix; x is an input vector; y is the hidden representation; b is an offset vector of dimensionality d'.

VI. Experimentation

We use two new real traffic datasets, namely the CSE-CIC-IDS2018 dataset [124] and the Bot-IoT dataset [122] for the experiments. Tables IV and V summarizes the statistics of attacks in Training and Test in both datasets. The experiment is performed on Google Colaboratory[1] under python 3 using TensorFlow and Graphics Processing Unit (GPU). The details

[1]https://colab.research.google.com

TABLE V: Attack Types in Bot-IoT dataset

Category	Attack Type	Flow Count	Training	Test
BENIGN	BENIGN	9543	7634	1909
Information gathering	Service scanning	1463364	117069	29267
	OS Fingerprinting	358275	28662	7166
DDoS attack	DDoS TCP	19547603	1563808	390952
	DDoS UDP	18965106	1517208	379302
	DDoS HTTP	19771	1582	395
DoS attack	DoS TCP	12315997	985280	246320
	DoS UDP	20659491	1652759	413190
	DoS HTTP	29706	2376	594
Information theft	Keylogging	1469	1175	294
	Data theft	118	94	24
Total	/	73370443	5877647	1469413

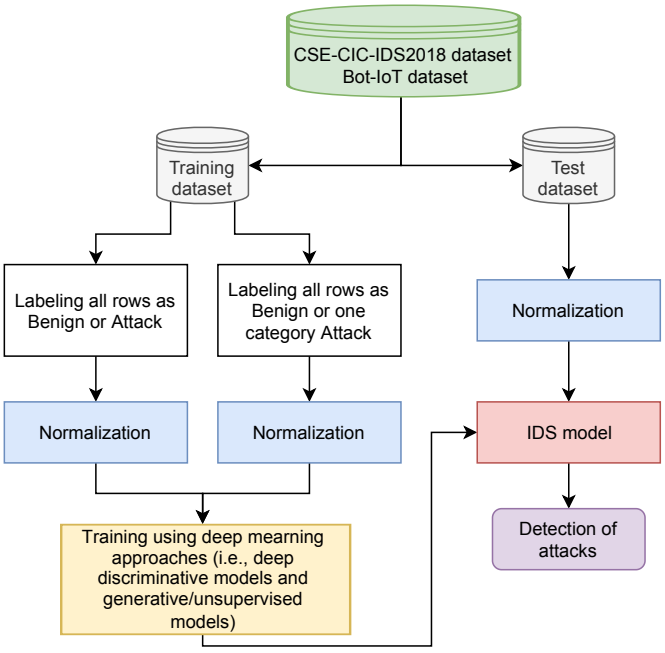


Fig. 10: Flowchart of the IDS methodology.

of the IDS methodology used in experimentation are illustrated in Fig. 10. Specifically, the method consists of four stages: 1) datasets stage, 2) pre-processing stage, 3) training stage and 4) testing stage. The hyperparameters used in deep learning approaches are presented in Table VI.

A. Data-set Pre-processing

The CSE-CIC-IDS2018 dataset contains 15450706 rows devised on 10 files, each row having 80 features. The contents of these files are described as following:

- File 1 "Wednesday-14-02-2018": It contains FTP-BruteForce (193360 rows), SSH-Bruteforce (187589 rows), and benign traffic (667626 rows).
- File 2 "Thursday-15-02-2018": It contains DoS attacks-GoldenEye (41508 rows), DoS attacks-Slowloris (10990 rows), and benign traffic (996077 rows).
- File 3 "Friday-16-02-2018": It contains DoS attacks-SlowHTTPTest (139890 rows), DoS attacks-Hulk (466664 rows), and benign traffic (442020 rows).

TABLE VI: The hyperparameters used in deep learning approaches

Hyperparameter	Value
Learning rate (LR)	0.01 - 0.5
Number of epoch	100
Hidden nodes (HN)	15 - 100
Batch size	1000
Classification function	SoftMax
Activation function	Sigmoid

TABLE VII: Confusion matrix

		Predicted class	
		Negative class	Positive class
Class	Negative class	True negative (TN)	False positive (FP)
	Positive class	False negative (FN)	True positive (TP)

- File 4 "Thursday-20-02-2018": It contains DDOS attack-LOIC-HTTP (576191 rows) and benign traffic (7372557 rows).
- File 5 "Wednesday-21-02-2018": It contains DDOS attack-LOIC-UDP (1730 rows), DDOS attack-HOIC (686012 rows), benign traffic (360833 rows).
- File 6 "Thursday-22-02-2018": It contains Brute Force -XSS (79 rows), Brute Force-Web (249 rows), SQL Injection (34 rows), and benign traffic (1048213 rows).
- File 7 "Friday-23-02-2018": It contains Brute Force -XSS (151 rows), Brute Force-Web (249 rows), SQL Injection (53 rows), benign traffic (1048009 rows).
- File 8 "Wednesday-28-02-2018": It contains Infiltration attack (68871 rows) and benign traffic (544200 rows).
- File 9 "Thursday-01-03-2018": It contains Infiltration attack (93063 rows) and benign traffic (238037 rows).
- File 10 "Friday-02-03-2018": It contains Botnet attack (286191 rows) and benign traffic (762384 rows).

The BoT-IoT dataset contains more than 72.000.000 records devised on 74 files, each row having 46 features. We use the version proposed by Koroniotis et al. [123], which is a version of training and testing with 5% of the entire dataset. In order to create a subset of training and testing, we import the files into one JSON document using PyMongo 3.7.2.

B. Performance metrics

We use the most important performance indicators, including, detection rate (DR), false alarm rate (FAR) and accuracy (ACC). Table VII shows the four possible cases of correct and wrong classification.

$$DR_{Attack} = \frac{TP_{Attack}}{TP_{Attack} + FN_{Attack}} \tag{12}$$

$$TNR_{BENIGN} = \frac{TN_{BENIGN}}{TN_{BENIGN} + FP_{BENIGN}} \tag{13}$$

$$FAR = \frac{FP_{BENIGN}}{TN_{BENIGN} + FP_{BENIGN}} \tag{14}$$

$$Accuracy = \frac{TP_{Attack} + TN_{BENIGN}}{TP_{Attack} + FN_{Attack} + TN_{BENIGN} + FP_{BENIGN}} \tag{15}$$

TABLE VIII: Performance of deep discriminative models relative to the different attack type and benign

	DNN	RNN	CNN
TNR (BENIGN)	96.915%	98.112%	98.914%
DR SSH-Bruteforce	100%	100%	100%
DR FTP-BruteForce	100%	100%	100%
DR Brute Force -XSS	83.265%	92.182%	92.101%
DR Brute Force -Web	82.223%	91.322%	91.002%
DR SQL Injection	100%	100%	100%
DR DoS attacks-Hulk	93.333%	94.912%	94.012%
DR DoS attacks-SlowHTTPTest	94.513%	96.123%	96.023%
DR DoS attacks-Slowloris	98.140%	98.220%	98.120%
DR DoS attacks-GoldenEye	92.110%	98.330%	98.221%
DR DDOS attack-HOIC	98.640%	98.711%	98.923%
DR DDOS attack-LOIC-UDP	97.348%	97.118%	97.888%
DR DDOS attack-LOIC-HTTP	97.222%	98.122%	98.991%
DR Botnet	96.420%	98.101%	98.982%
DR Infilteration	97.518%	97.874%	97.762%
DR Service scanning	96.428%	96.874%	97.102%
DR OS Fingerprinting	96.139%	96.762%	97.001%
DR DDoS TCP	96.219%	96.650%	97.003%
DR DDoS UDP	96.118%	96.666%	97.006%
DR DDoS HTTP	96.616%	96.564%	97.010%
DR DoS TCP	96.628%	96.772%	97.110%
DR DoS UDP	96.525%	96.761%	97.112%
DR DoS HTTP	96.699%	96.868%	97.512%
DR Keylogging	96.762%	96.999%	98.102%
DR Data theft	100%	100%	100%

$$DR_{Overall} = \frac{\sum TP_{Each-Attack-Type}}{\sum TP_{Each-Attack-Type} + \sum FN_{Each-Attack-Type}} \quad (16)$$

where TP, TN, FP, and FN denote true positive, true negative, false positive, and false negative, respectively.

C. Results

Table VIII shows the performance of deep discriminative models relative to the different attack types and benign. It shows that deep neural network gives the highest true negative rate with 96.915%. The recurrent neural network gives the highest detection rate for seven attack types, namely, Brute Force -XSS 92.182%, Brute Force -Web 91.322%, DoS attacks-Hulk 94.912%, DoS attacks-SlowHTTPTest 96.123%, DoS attacks-Slowloris 98.220%, DoS attacks-GoldenEye 98.330%, and Infilteration 97.874%. The convolutional neural network gives the higest detection rate for four attacks type, including, DDOS attack-HOIC 98.923%, DDOS attack-LOIC-UDP 97.888%, and DDOS attack-LOIC-HTTP 98.991%, and Botnet 98.982%.

The performance of generative/unsupervised models relative to the different attack types and benign is shown in Table IX. It can be seen that deep belief network gives the highest true negative rate with 98.212% and the higest detection rate for four attacks type, namely, Brute Force -XSS 92.281%, Brute Force -Web 91.427%, DoS attacks-Hulk 91.712%, and

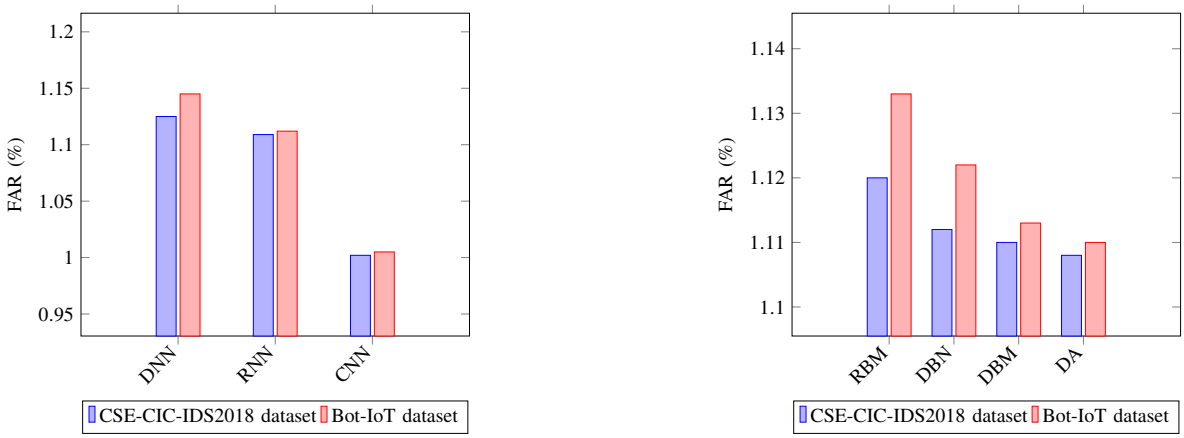
TABLE IX: Performance of generative/unsupervised models relative to the different attack type and benign

	RBM	DBN	DBM	DA
TNR (BENIGN)	97.316%	98.212%	96.215%	98.101%
DR SSH-Bruteforce	100%	100%	100%	100%
DR FTP-BruteForce	100%	100%	100%	100%
DR Brute Force -XSS	83.164%	92.281%	92.103%	95.223%
DR Brute Force -Web	82.221%	91.427%	91.254%	95.311%
DR SQL Injection	100%	100%	100%	100%
DR DoS attacks-Hulk	91.323%	91.712%	93.072%	92.112%
DR DoS attacks-SlowHTTPTest	93.313%	95.273%	95.993%	94.191%
DR DoS attacks-Slowloris	97.040%	97.010%	97.112%	97.120%
DR DoS attacks-GoldenEye	92.010%	97.130%	97.421%	96.222%
DR DDOS attack-HOIC	97.541%	97.211%	97.121%	96.551%
DR DDOS attack-LOIC-UDP	96.148%	96.122%	96.654%	96.445%
DR DDOS attack-LOIC-HTTP	96.178%	97.612%	97.121%	97.102%
DR Botnet	96.188%	97.221%	97.812%	97.717%
DR Infilteration	96.411%	96.712%	96.168%	97.818%
DR Service scanning	96.301%	96.602%	96.067%	97.712%
DR OS Fingerprinting	96.302%	96.606%	96.077%	97.715%
DR DDoS TCP	96.512%	96.602%	96.075%	97.712%
DR DDoS UDP	96.522%	96.623%	96.111%	97.989%
DR DDoS HTTP	96.544%	96.721%	96.214%	97.991%
DR DoS TCP	96.567%	96.724%	96.333%	97.995%
DR DoS UDP	96.561%	96.828%	96.654%	98.031%
DR DoS HTTP	96.799%	96.911%	96.994%	98.412%
DR Keylogging	97.112%	97.662%	98.224%	98.331%
DR Data theft	100%	100%	100%	100%

TABLE X: The accuracy and training time of deep discriminative models with different learning rate and hidden nodes in the CSE-CIC-IDS2018 dataset

Parameters	Accuracy and training time (s)	DNN	RNN	CNN
HN = 15	ACC	96.552%	96.872%	96.915%
LR=0.01	Time	20.2	30.3	28.4
HN = 15	ACC	96.651%	96.882%	96.912%
LR=0.1	Time	19.1	29.2	27.2
HN = 15	ACC	96.653%	96.886%	96.913%
LR=0.5	Time	18.9	29.1	27.1
HN = 30	ACC	96.612%	96.881%	96.922%
LR=0.01	Time	88.1	91.3	89.6
HN = 30	ACC	96.658%	96.888%	96.926%
LR=0.1	Time	87.9	90.9	88.5
HN = 30	ACC	96.662%	96.891%	96.929%
LR=0.5	Time	86.1	90.3	87.9
HN = 60	ACC	96.701%	96.903%	96.922%
LR=0.01	Time	180.2	197.5	192.2
HN = 60	ACC	96.921%	96.970%	96.975%
LR=0.1	Time	179.3	192.2	189.1
HN = 60	ACC	96.950%	96.961%	96.992%
LR=0.5	Time	177.7	190.6	182.6
HN = 100	ACC	97.102%	97.111%	97.222%
LR=0.01	Time	395.2	341.5	338.9
HN = 100	ACC	97.187%	97.229%	97.312%
LR=0.1	Time	391.1	336.9	332.5
HN = 100	ACC	97.281%	97.310%	97.376%
LR=0.5	Time	390.2	334.7	331.2

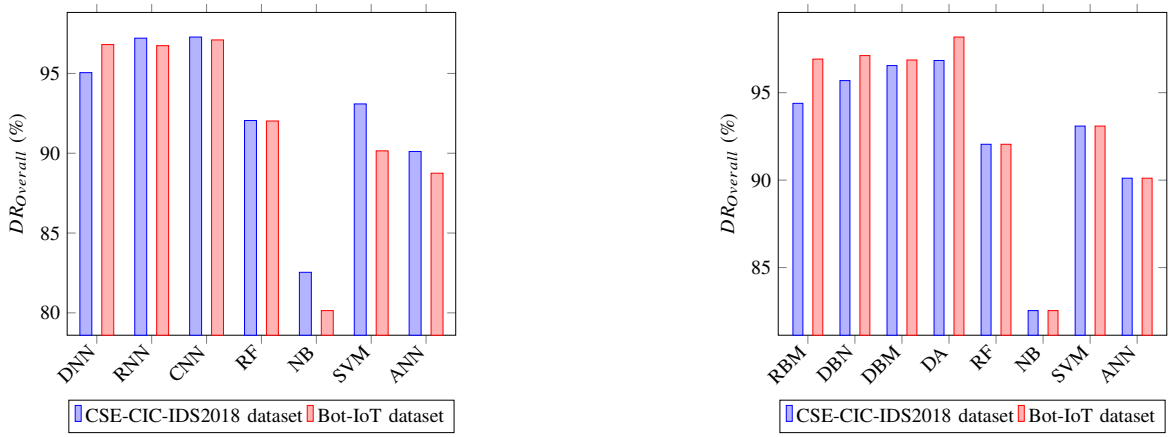
DDOS attack-LOIC-HTTP 97.612%. The deep auto encoders give the higest detection rate for three attack types, namely, Brute Force -Web 95.311%, DoS attacks-Slowloris 97.120%, and Infilteration 97.818%. The deep Boltzmann machine gives



(a) Deep discriminative models

(b) Generative/unsupervised models

Fig. 11: Performance of deep learning approaches in term of false alarm rate



(a) Deep discriminative models

(b) Generative/unsupervised models

Fig. 12: Performance of deep learning approaches compared with other machine learning approaches in term of global detection rate. RF: Random forests, NB: Naive Bayes, SVM: Support Vector Machine, ANN: Artificial neural network.

the higest detection rate for five attack types, namely, DoS attacks-Hulk 93.072%, DoS attacks-SlowHTTPTest 95.993%, DoS attacks-GoldenEye 97.421%, DDOS attack-LOIC-UDP 96.654%, and Botnet 97.812%.

Table X presents the accuracy and training time of deep discriminative models with different learning rates and hidden nodes in the CSE-CIC-IDS2018 dataset. Compared to both deep neural network and recurrent neural network, the convolutional neural network gets a higher accuracy 97.376%, when there are 100 hidden nodes and the learning rate is 0.5. Table XI presents the accuracy and training time of deep discriminative models in the Bot-IoT dataset with different learning rate and hidden nodes. The convolutional neural network gets a higher accuracy 98.371%, when there are 100 hidden nodes and the learning rate is 0.5. In addition, the training time of deep neural network is always less than others related techniques (i.e., recurrent neural network and convolutional neural network).

Table XII demonstrates the accuracy and training time of generative/unsupervised models in the CSE-CIC-IDS2018 dataset with different learning rates and hidden nodes. The deep auto encoders gets a higher accuracy 97.372%, when there are 100 hidden nodes and the learning rate is 0.5 compared to three techniques, including, restricted Boltzmann machine, deep belief network, and deep boltzmann machine. Table XIII demonstrates the accuracy and training time of generative/unsupervised models in the Bot-IoT dataset with different learning rate and hidden nodes. The deep auto encoders gets a higher accuracy 98.394%, when there are 100 hidden nodes and the learning rate is 0.5. In addition, the training time of restricted Boltzmann machine is always less than others related techniques (i.e., deep belief network, deep boltzmann machine, and deep auto encoders).

The performance of deep learning approaches in term of false alarm rate in CSE-CIC-IDS2018 dataset and Bot-IoT dataset is depicted in Figure 11. In the generative/unsupervised models, mean false alarm rate of the convolutional neural network is better than both deep neural network and recurrent neural network. In the deep discriminative models, mean false alarm rate of the deep autoencoders is better than three

TABLE XI: The accuracy and training time of deep discriminative models with different learning rate and hidden nodes in the Bot-IoT dataset

Parameters	Accuracy and training time (s)	DNN	RNN	CNN
HN = 15	ACC	96.446%	96.765%	96.900%
LR=0.01	Time	56.5	70.7	65.3
HN = 15	ACC	96.651%	96.882%	96.912%
LR=0.1	Time	66.6	92.6	91.3
HN = 15	ACC	96.651%	96.884%	96.910%
LR=0.5	Time	88.1	102.5	101.1
HN = 30	ACC	96.611%	96.877%	96.919%
LR=0.01	Time	88.1	102.5	101.1
HN = 30	ACC	96.655%	96.882%	96.921%
LR=0.1	Time	102.2	150.4	144.2
HN = 30	ACC	96.661%	96.898%	97.101%
LR=0.5	Time	170.3	222.1	221.7
HN = 60	ACC	96.766%	96.955%	97.102%
LR=0.01	Time	250.8	331.2	339.6
HN = 60	ACC	96.922%	96.974%	97.212%
LR=0.1	Time	302.9	377.1	366.2
HN = 60	ACC	97.102%	97.291%	97.881%
LR=0.5	Time	391.1	451.2	412.2
HN = 100	ACC	97.221%	97.618%	97.991%
LR=0.01	Time	600.2	801.5	812.2
HN = 100	ACC	97.501%	97.991%	98.121%
LR=0.1	Time	711.9	1001.8	1022.1
HN = 100	ACC	98.221%	98.311%	98.371%
LR=0.5	Time	991.6	1400.6	1367.2

TABLE XII: The accuracy and training time of generative/unsupervised models with different learning rate and hidden nodes in the CSE-CIC-IDS2018 dataset

Parameters	Accuracy and training time (s)	RBM	DBN	DBM	DA
HN = 15	ACC	96.551%	96.852%	96.911%	96.912%
LR=0.01	Time	20.0	30.1	28.3	28.3
HN = 15	ACC	96.642%	96.871%	96.901%	96.902%
LR=0.1	Time	19.0	29.1	27.1	27.2
HN = 15	ACC	96.651%	96.885%	96.910%	96.911%
LR=0.5	Time	18.8	28.1	26.2	27.1
HN = 30	ACC	96.602%	96.844%	96.918%	96.917%
LR=0.01	Time	88.0	90.4	89.5	88.6
HN = 30	ACC	96.656%	96.884%	96.922%	96.923%
LR=0.1	Time	87.4	90.7	88.3	88.2
HN = 30	ACC	96.661%	96.890%	96.925%	96.924%
LR=0.5	Time	86.1	90.3	87.9	87.10
HN = 60	ACC	96.691%	96.883%	96.912%	96.913%
LR=0.01	Time	180.1	196.5	191.1	191.4
HN = 60	ACC	96.920%	96.967%	96.972%	96.971%
LR=0.1	Time	179.1	192.1	189.0	189.1
HN = 60	ACC	96.947%	96.960%	96.991%	96.992%
LR=0.5	Time	177.6	190.5	181.4	181.4
HN = 100	ACC	97.101%	97.108%	97.211%	97.221%
LR=0.01	Time	394.1	340.4	339.1	337.11
HN = 100	ACC	97.186%	97.227%	97.300%	97.311%
LR=0.1	Time	390.0	334.8	330.1	331.7
HN = 100	ACC	97.280%	97.302%	97.371%	97.372%
LR=0.5	Time	390.1	344.7	351.5	341.3

TABLE XIII: The accuracy and training time of generative/unsupervised models with different learning rates and hidden nodes in the Bot-IoT dataset

Parameters	Accuracy and training time (s)	RBM	DBN	DBM	DA
HN = 15	ACC	96.652%	96.551%	96.411%	96.717%
LR=0.01	Time	50.4	72.8	60.2	60.1
HN = 15	ACC	96.666%	96.882%	96.922%	96.934%
LR=0.1	Time	100.2	138.2	133.1	133.7
HN = 15	ACC	96.655%	96.892%	96.914%	96.955%
LR=0.5	Time	150.5	221.7	201.9	210.3
HN = 30	ACC	96.616%	96.862%	96.940%	96.960%
LR=0.01	Time	400.8	560.2	522.1	524.2
HN = 30	ACC	96.756%	96.924%	97.911%	97.923%
LR=0.1	Time	701.6	801.1	788.1	791.6
HN = 30	ACC	96.755%	96.990%	97.925%	97.924%
LR=0.5	Time	1022.6	1291.6	1239.6	1266.8
HN = 60	ACC	96.871%	97.183%	97.922%	97.927%
LR=0.01	Time	1129.6	1461.6	1432.6	1461.2
HN = 60	ACC	97.221%	97.961%	97.971%	97.996%
LR=0.1	Time	1421.1	1912.8	1811.9	1821.1
HN = 60	ACC	97.722%	97.981%	97.998%	98.001%
LR=0.5	Time	1771.9	2201.9	2109.8	2101.8
HN = 100	ACC	98.201%	98.107%	98.312%	98.322%
LR=0.01	Time	1861.7	2521.8	2401.1	2466.2
HN = 100	ACC	98.214%	98.122%	98.371%	98.312%
LR=0.1	Time	1991.6	2644.2	2531.2	2566.9
HN = 100	ACC	98.281%	98.312%	98.381%	98.394%
LR=0.5	Time	2111.9	2921.7	2800.1	2816.2

techniques, including, restricted Boltzmann machine, deep belief network, and deep Boltzmann machine.

Figure 9 presents the performance of deep learning approaches compared with four machine learning approaches, including, Naive Bayes, Artificial neural network, Support Vector Machine, and Random forests., in term of global detection rate $DR_{Overall}$. In the deep discriminative models, the CNN model gives the highest overall detection rate (DR Overall) with 97.28% in CSE-CIC-IDS2018 dataset and 97,01% in Bot-IoT dataset compared to RF, NB, SVM, and ANN. In the generative/unsupervised models, the DA model gives the highest overall detection rate (DR Overall) with 98.18% in CSE-CIC-IDS2018 dataset and 96.84% in Bot-IoT dataset compared to RF, NB, SVM, and ANN.

VII. Conclusion

In this paper, we conducted a comparative study of deep learning approaches for intrusion detection, namely, deep discriminative models and generative/unsupervised models. Specifically, we analyzed seven deep learning approaches, including recurrent neural networks, deep neural networks, restricted Boltzmann machine, deep belief networks, convolutional neural networks, deep Boltzmann machines, and deep autoencoders. These machine learning methods are compared using two new datasets, the CSE-CIC-IDS2018 dataset and the Bot-IoT dataset with three important performance indicators, namely, false alarm rate, accuracy, and detection rate.

REFERENCES

[1] L. A. Maglaras, K.-H. Kim, H. Janicke, M. A. Ferrag, S. Rallis, P. Fragkou, A. Maglaras, and T. J. Cruz, "Cyber security of critical infrastructures," *ICT Express*, vol. 4, no. 1, pp. 42–45, 2018.

[2] A. Ahmim, M. Derdour, and M. A. Ferrag, "An intrusion detection system based on combining probability predictions of a tree of classifiers," *International Journal of Communication Systems*, vol. 31, no. 9, p. e3547, 2018.

[3] A. Ahmim, L. Maglaras, M. A. Ferrag, M. Derdour, and H. Janicke, "A novel hierarchical intrusion detection system based on decision tree and rules-based models," *arXiv preprint arXiv:1812.09059*, 2018.

[4] Z. Dewa and L. A. Maglaras, "Data mining and intrusion detection systems," *International Journal of Advanced Computer Science and Applications*, vol. 7, no. 1, pp. 62–71, 2016.

[5] B. Stewart, L. Rosa, L. A. Maglaras, T. J. Cruz, M. A. Ferrag, P. Simões, and H. Janicke, "A novel intrusion detection mechanism for scada systems which automatically adapts to network topology changes." *EAI Endorsed Trans. Indust. Netw. & Intellig. Syst.*, vol. 4, no. 10, p. e4, 2017.

[6] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization." in *ICISSP*, 2018, pp. 108–116.

[7] M. A. Ferrag, L. Maglaras, H. Janicke, and R. Smith, "Deep learning techniques for cyber security intrusion detection : A detailed analysis," in *6th International Symposium for ICS & SCADA Cyber Security Research (ICS-CSR 2019), Athens, 10-12 September*, 2019.

[8] A. L. Buczak and E. Guven, "A survey of data mining and machine learning methods for cyber security intrusion detection," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 2, pp. 1153–1176, 2015.

[9] A. Milenkoski, M. Vieira, S. Kounev, A. Avritzer, and B. D. Payne, "Evaluating computer intrusion detection systems: A survey of common practices," *ACM Computing Surveys (CSUR)*, vol. 48, no. 1, p. 12, 2015.

[10] G. Folino and P. Sabatino, "Ensemble based collaborative and distributed intrusion detection systems: A survey," *Journal of Network and Computer Applications*, vol. 66, pp. 1–16, 2016.

[11] B. B. Zarpelao, R. S. Miani, C. T. Kawakani, and S. C. de Alvarenga, "A survey of intrusion detection in internet of things," *Journal of Network and Computer Applications*, vol. 84, pp. 25–37, 2017.

[12] A. A. Aburomman and M. B. I. Reaz, "A survey of intrusion detection systems based on ensemble and hybrid classifiers," *Computers & Security*, vol. 65, pp. 135–152, 2017.

[13] Y. Xin, L. Kong, Z. Liu, Y. Chen, Y. Li, H. Zhu, M. Gao, H. Hou, and C. Wang, "Machine learning and deep learning methods for cybersecurity," *IEEE Access*, vol. 6, pp. 35 365–35 381, 2018.

[14] M. Ring, S. Wunderlich, D. Scheuring, D. Landes, and A. Hotho, "A survey of network-based intrusion detection data sets," *Computers & Security*, 2019.

[15] G. Loukas, E. Karapistoli, E. Panaousis, P. Sarigiannidis, A. Bezemskij, and T. Vuong, "A taxonomy and survey of cyber-physical intrusion detection approaches for vehicles," *Ad Hoc Networks*, vol. 84, pp. 124–147, 2019.

[16] K. A. da Costa, J. P. Papa, C. O. Lisboa, R. Munoz, and V. H. C. de Albuquerque, "Internet of things: A survey on machine learning-based intrusion detection approaches," *Computer Networks*, vol. 151, pp. 147–157, 2019.

[17] N. Chaabouni, M. Mosbah, A. Zemmari, C. Sauvignac, and P. Faruki, "Network intrusion detection for iot security based on learning techniques," *IEEE Communications Surveys & Tutorials*, 2019.

[18] D. S. Berman, A. L. Buczak, J. S. Chavis, and C. L. Corbett, "A survey of deep learning methods for cyber security," *Information*, vol. 10, no. 4, p. 122, 2019.

[19] S. Mahdavifar and A. A. Ghorbani, "Application of deep learning to cybersecurity: A survey," *Neurocomputing*, 2019.

[20] N. Sultana, N. Chilamkurti, W. Peng, and R. Alhadad, "Survey on sdn based network intrusion detection system using machine learning approaches," *Peer-to-Peer Networking and Applications*, vol. 12, no. 2, pp. 493–501, 2019.

[21] T. A. Tang, L. Mhamdi, D. McLernon, S. A. R. Zaidi, and M. Ghogho, "Deep learning approach for network intrusion detection in software defined networking," in *2016 International Conference on Wireless Networks and Mobile Communications (WINCOM)*. IEEE, 2016, pp. 258–263.

[22] S. Potluri and C. Diedrich, "Accelerated deep neural networks for enhanced intrusion detection system," in *2016 IEEE 21st International Conference on Emerging Technologies and Factory Automation (ETFA)*. IEEE, 2016, pp. 1–8.

[23] M.-J. Kang and J.-W. Kang, "Intrusion detection system using deep neural network for in-vehicle network security," *PloS one*, vol. 11, no. 6, p. e0155781, 2016.

[24] L. Zhou, X. Ouyang, H. Ying, L. Han, Y. Cheng, and T. Zhang, "Cyber-attack classification in smart grid via deep neural network," in *Proceedings of the 2nd International Conference on Computer Science and Application Engineering*. ACM, 2018, p. 90.

[25] F. Feng, X. Liu, B. Yong, R. Zhou, and Q. Zhou, "Anomaly detection in ad-hoc networks based on deep learning model: A plug and play device," *Ad Hoc Networks*, vol. 84, pp. 82–89, 2019.

[26] H. Zhang, X. Yu, P. Ren, C. Luo, and G. Min, "Deep adversarial learning in intrusion detection: A data augmentation enhanced framework," *arXiv preprint arXiv:1901.07949*, 2019.

[27] S. S. Roy, A. Mallik, R. Gulati, M. S. Obaidat, and P. Krishna, "A deep learning based artificial neural network approach for intrusion detection," in *International Conference on Mathematics and Computing*. Springer, 2017, pp. 44–53.

[28] J. Kim, N. Shin, S. Y. Jo, and S. H. Kim, "Method of intrusion detection using deep neural network," in *2017 IEEE International Conference on Big Data and Smart Computing (BigComp)*. IEEE, 2017, pp. 313–316.

[29] L. Zhang, L. Shi, N. Kaja, and D. Ma, "A two-stage deep learning approach for can intrusion detection," in *Proc. Ground Vehicle Syst. Eng. Technol. Symp.(GVSETS)*, 2018, pp. 1–11.

[30] S. M. Kasongo and Y. Sun, "A deep learning method with filter based feature engineering for wireless intrusion detection system," *IEEE Access*, vol. 7, pp. 38 597–38 607, 2019.

[31] J. Kim, J. Kim, H. L. T. Thu, and H. Kim, "Long short term memory recurrent neural network classifier for intrusion detection," in *2016 International Conference on Platform Technology and Service (PlatCon)*. IEEE, 2016, pp. 1–5.

[32] A. Taylor, S. Leblanc, and N. Japkowicz, "Anomaly detection in automobile control network data with long short-term memory networks," in *2016 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*. IEEE, 2016, pp. 130–139.

[33] G. Loukas, T. Vuong, R. Heartfield, G. Sakellari, Y. Yoon, and D. Gan, "Cloud-based cyber-physical intrusion detection for vehicles using deep learning," *Ieee Access*, vol. 6, pp. 3491–3508, 2017.

[34] C. Yin, Y. Zhu, J. Fei, and X. He, "A deep learning approach for intrusion detection using recurrent neural networks," *Ieee Access*, vol. 5, pp. 21 954–21 961, 2017.

[35] T. A. Tang, L. Mhamdi, D. McLernon, S. A. R. Zaidi, and M. Ghogho, "Deep recurrent neural network for intrusion detection in sdn-based networks," in *2018 4th IEEE Conference on Network Softwarization and Workshops (NetSoft)*. IEEE, 2018, pp. 202–206.

[36] F. Jiang, Y. Fu, B. B. Gupta, F. Lou, S. Rho, F. Meng, and Z. Tian, "Deep learning based multi-channel intelligent attack detection for data security," *IEEE Transactions on Sustainable Computing*, 2018.

[37] M. A. Ferrag and L. Maglaras, "Deepcoin: A novel deep learning and blockchain-based energy exchange framework for smart grids," *IEEE Transactions on Engineering Management*, 2019.

[38] S. Basumallik, R. Ma, and S. Eftekharnejad, "Packet-data anomaly detection in pmu-based state estimator using convolutional neural network," *International Journal of Electrical Power & Energy Systems*, vol. 107, pp. 690–702, 2019.

[39] K. Fu, D. Cheng, Y. Tu, and L. Zhang, "Credit card fraud detection using convolutional neural networks," in *International Conference on Neural Information Processing*. Springer, 2016, pp. 483–490.

[40] Z. Zhang, X. Zhou, X. Zhang, L. Wang, and P. Wang, "A model based on convolutional neural network for online transaction fraud detection," *Security and Communication Networks*, vol. 2018, 2018.

[41] M. Nasr, A. Bahramali, and A. Houmansadr, "Deepcorr: Strong flow correlation attacks on tor using deep learning," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2018, pp. 1962–1976.

[42] Y. Zhang, X. Chen, L. Jin, X. Wang, and D. Guo, "Network intrusion detection: Based on deep hierarchical network and original flow data," *IEEE Access*, vol. 7, pp. 37 004–37 016, 2019.

[43] Y. Zeng, H. Gu, W. Wei, and Y. Guo, "$deep-full-range$: A deep learning based network encrypted traffic classification and intrusion detection framework," *IEEE Access*, vol. 7, pp. 45 182–45 190, 2019.

[44] Y. Yu, J. Long, and Z. Cai, "Network intrusion detection through stacking dilated convolutional autoencoders," *Security and Communication Networks*, vol. 2017, 2017.

[45] K. Alrawashdeh and C. Purdy, "Toward an online anomaly intrusion detection system based on deep learning," in *2016 15th IEEE International Conference on Machine Learning and Applications (ICMLA)*. IEEE, 2016, pp. 195–200.

[46] T. Aldwairi, D. Perera, and M. A. Novotny, "An evaluation of the performance of restricted boltzmann machines as a model for anomaly network intrusion detection," *Computer Networks*, vol. 144, pp. 111–119, 2018.

[47] U. Fiore, F. Palmieri, A. Castiglione, and A. De Santis, "Network anomaly detection with the restricted boltzmann machine," *Neurocomputing*, vol. 122, pp. 13–23, 2013.

[48] M. A. Salama, H. F. Eid, R. A. Ramadan, A. Darwish, and A. E. Hassanien, "Hybrid intelligent intrusion detection scheme," in *Soft computing in industrial applications*. Springer, 2011, pp. 293–303.

[49] N. Gao, L. Gao, Q. Gao, and H. Wang, "An intrusion detection model based on deep belief networks," in *2014 Second International Conference on Advanced Cloud and Big Data*. IEEE, 2014, pp. 247–252.

[50] M. Z. Alom, V. Bontupalli, and T. M. Taha, "Intrusion detection using deep belief networks," in *2015 National Aerospace and Electronics Conference (NAECON)*. IEEE, 2015, pp. 339–344.

[51] J. Yang, J. Deng, S. Li, and Y. Hao, "Improved traffic detection with support vector machine based on restricted boltzmann machine," *Soft Computing*, vol. 21, no. 11, pp. 3101–3112, 2017.

[52] S. Otoum, B. Kantarci, and H. T. Mouftah, "On the feasibility of deep learning in sensor network intrusion detection," *IEEE Networking Letters*, 2019.

[53] H. Karimipour, A. Dehghantanha, R. Parizi, K. Choo, and H. Leung, "A deep and scalable unsupervised machine learning system for cyber-attack detection in large-scale smart grids," *IEEE Access*, 2019.

[54] G. Thamilarasu and S. Chawla, "Towards deep-learning-driven intrusion detection for the internet of things," *Sensors*, vol. 19, no. 9, p. 1977, 2019.

[55] G. Zhao, C. Zhang, and L. Zheng, "Intrusion detection using deep belief network and probabilistic neural network," in *2017 IEEE International Conference on Computational Science and Engineering (CSE) and IEEE International Conference on Embedded and Ubiquitous Computing (EUC)*, vol. 1. IEEE, 2017, pp. 639–642.

[56] Y. Zhang, P. Li, and X. Wang, "Intrusion detection for iot based on improved genetic algorithm and deep belief network," *IEEE Access*, vol. 7, pp. 31 711–31 722, 2019.

[57] M. Aloqaily, S. Otoum, I. Al Ridhawi, and Y. Jararweh, "An intrusion detection system for connected vehicles in smart cities," *Ad Hoc Networks*, vol. 90, p. 101842, 2019.

[58] Y. He, G. J. Mendis, and J. Wei, "Real-time detection of false data injection attacks in smart grid: A deep learning-based intelligent mechanism," *IEEE Transactions on Smart Grid*, vol. 8, no. 5, pp. 2505–2516, 2017.

[59] N. Shone, T. N. Ngoc, V. D. Phai, and Q. Shi, "A deep learning approach to network intrusion detection," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 2, no. 1, pp. 41–50, 2018.

[60] F. A. Khan, A. Gumaei, A. Derhab, and A. Hussain, "Tsdl: A twostage deep learning model for efficient network intrusion detection," *IEEE Access*, 2019.

[61] D. Papamartzivanos, F. G. Mármol, and G. Kambourakis, "Introducing deep learning self-adaptive misuse network intrusion detection systems," *IEEE Access*, vol. 7, pp. 13 546–13 560, 2019.

[62] Y. Yang, K. Zheng, C. Wu, and Y. Yang, "Improving the classification effectiveness of intrusion detection by using improved conditional variational autoencoder and deep neural network," *Sensors*, vol. 19, no. 11, p. 2528, 2019.

[63] A. Abusitta, M. Bellaiche, M. Dagenais, and T. Halabi, "A deep learning approach for proactive multi-cloud cooperative intrusion detection system," *Future Generation Computer Systems*, 2019.

[64] Y. Wang, W.-d. Cai, and P.-c. Wei, "A deep learning approach for detecting malicious javascript code," *security and communication networks*, vol. 9, no. 11, pp. 1520–1534, 2016.

[65] D. Li, L. Deng, M. Lee, and H. Wang, "Iot data feature extraction and intrusion detection system for smart cities based on deep migration learning," *International Journal of Information Management*, 2019.

[66] A. Javaid, Q. Niyaz, W. Sun, and M. Alam, "A deep learning approach for network intrusion detection system," in *Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies*, 2016, pp. 21–26.

[67] C. G. Cordero, S. Hauke, M. Mühlhäuser, and M. Fischer, "Analyzing flow-based anomaly intrusion detection using replicator neural networks," in *2016 14th Annual Conference on Privacy, Security and Trust (PST)*. IEEE, 2016, pp. 317–324.

[68] "Cicds2017 dataset," https://www.unb.ca/cic/datasets/ids-2017.html. last accessed 30 May 2019.

[69] "Iscx dataset," https://www.unb.ca/cic/datasets/ids.html. last accessed 23 Juin 2019.

[70] S. Otoum, B. Kantarci, and H. Mouftah, "Adaptively supervised and intrusion-aware data aggregation for wireless sensor clusters in critical infrastructures," in *2018 IEEE International Conference on Communications (ICC)*. IEEE, 2018, pp. 1–6.

[71] O. Y. Al-Jarrah, C. Maple, M. Dianati, D. Oxtoby, and A. Mouzakitis, "Intrusion detection systems for intra-vehicle networks: A review," *IEEE Access*, vol. 7, pp. 21 266–21 289, 2019.

[72] G. W. Taylor, G. E. Hinton, and S. T. Roweis, "Modeling human motion using binary latent variables," in *Advances in neural information processing systems*, 2007, pp. 1345–1352.

[73] "1998 darpa intrusion detection evaluation," https://www.ll.mit.edu/r-d/datasets/1998-darpa-intrusion-detection-evaluation-dataset. last accessed 30 May 2019.

[74] R. P. Lippmann, D. J. Fried, I. Graf, J. W. Haines, K. R. Kendall, D. McClung, D. Weber, S. E. Webster, D. Wyschogrod, R. K. Cunningham *et al.*, "Evaluating intrusion detection systems: The 1998 darpa off-line intrusion detection evaluation," in *Proceedings DARPA Information Survivability Conference and Exposition. DISCEX'00*, vol. 2. IEEE, 2000, pp. 12–26.

[75] "Kdd cup 1999," http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html. last accessed 30 May 2019.

[76] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the kdd cup 99 data set," in *2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications*. IEEE, 2009, pp. 1–6.

[77] "Nsl kdd," https://www.unb.ca/cic/datasets/nsl.html. last accessed 30 May 2019.

[78] "Unsw-nb15 dataset," https://www.unsw.adfa.edu.au/unsw-canberra-cyber/cybersecurity/ADFA-NB15-Datasets/. last accessed 30 May 2019.

[79] N. Moustafa and J. Slay, "Unsw-nb15: a comprehensive data set for network intrusion detection systems (unsw-nb15 network data set)," in *2015 military communications and information systems conference (MilCIS)*. IEEE, 2015, pp. 1–6.

[80] J. O. Nehinbe, "A simple method for improving intrusion detections in corporate networks," in *International Conference on Information Security and Digital Forensics*. Springer, 2009, pp. 111–122.

[81] "Center for applied internet data analysis," https://www.caida.org/data/overview/. last accessed 30 May 2019.

[82] M. Jonker, A. King, J. Krupp, C. Rossow, A. Sperotto, and A. Dainotti, "Millions of targets under attack: a macroscopic characterization of the dos ecosystem," in *Proceedings of the 2017 Internet Measurement Conference*. ACM, 2017, pp. 100–113.

[83] "Lbnl dataset," https://powerdata.lbl.gov/download.html. last accessed 23 Juin 2019.

[84] E. Stewart, A. Liao, and C. Roberts, "Open μpmu: A real world reference distribution micro-phasor measurement unit data set for research and application development," 2016.

[85] U. Adhikari, S. Pan, T. Morris, R. Borges, and J. Beave, "Industrial control system (ICS) cyber attack datasets," https://sites.google.com/a/uah.edu/tommy-morris-uah/ics-data-sets. last accessed 23 Juin 2019.

[86] S. Pan, T. Morris, and U. Adhikari, "Developing a hybrid intrusion detection system using data mining for power systems," *IEEE Transactions on Smart Grid*, vol. 6, no. 6, pp. 3104–3113, 2015.

[87] P. Hines, S. Blumsack, E. C. Sanchez, and C. Barrows, "The topological and electrical structure of power grids," in *2010 43rd Hawaii International Conference on System Sciences*. IEEE, 2010, pp. 1–10.

[88] "Cdx dataset," http://www.fit.vutbr.cz/~ihomoliak/asnm/ASNM-CDX-2009.html. last accessed 30 May 2019.

[89] I. Homoliak, M. Barabas, P. Chmelar, M. Drozd, and P. Hanacek, "Asnm: Advanced security network metrics for attack vector description," in *Proceedings of the International Conference on Security and Management (SAM)*. The Steering Committee of The World Congress in Computer Science, Computer âĂę, 2013, p. 1.

[90] "Kyoto dataset," http://www.takakura.com/Kyoto_data/. last accessed 30 May 2019.

[91] J. Song, H. Takakura, and Y. Okabe, "Description of kyoto university benchmark data," *Available at link: http://www. takakura. com/Kyoto_data/BenchmarkData-Description-v5. pdf [Accessed on 15 March 2016]*, 2006.

[92] "Mawi dataset," http://www.fukuda-lab.org/mawilab/data.html. last accessed 30 May 2019.

[93] R. Fontugne, P. Borgnat, P. Abry, and K. Fukuda, "Mawilab: combining diverse anomaly detectors for automated anomaly labeling and performance benchmarking," in *Proceedings of the 6th International COnference*. ACM, 2010, p. 8.

[94] "Heritrix dataset," http://crawler.archive.org/index.html. last accessed 30 May 2019.

[95] "Heritrix user manual," http://crawler.archive.org/articles/user_manual/index.html. last accessed 30 May 2019.

[96] "Android validation dataset," https://www.unb.ca/cic/datasets/android-validation.html. last accessed 30 May 2019.

[97] M. Shoaib, S. Bosch, O. Incel, H. Scholten, and P. Havinga, "Fusion of smartphone motion sensors for physical activity recognition," *Sensors*, vol. 14, no. 6, pp. 10 146–10 176, 2014.

[98] "Umass dataset," http://traces.cs.umass.edu. last accessed 23 Juin 2019.

[99] A. Shiravi, H. Shiravi, M. Tavallaee, and A. A. Ghorbani, "Toward developing a systematic approach to generate benchmark datasets for intrusion detection," *computers & security*, vol. 31, no. 3, pp. 357–374, 2012.

[100] "Adfa intrusion detection datasets," https://www.unsw.adfa.edu.au/unsw-canberra-cyber/cybersecurity/ADFA-IDS-Datasets/. last accessed 23 Juin 2019.

[101] G. Creech and J. Hu, "A semantic approach to host-based intrusion detection systems using contiguousand discontiguous system call patterns," *IEEE Transactions on Computers*, vol. 63, no. 4, pp. 807–819, 2013.

[102] "Vpn-nonvpn dataset," https://www.unb.ca/cic/datasets/vpn.html. last accessed 23 Juin 2019.

[103] G. Draper-Gil, A. H. Lashkari, M. S. I. Mamun, and A. A. Ghorbani, "Characterization of encrypted and vpn traffic using time-related," in *Proceedings of the 2nd international conference on information systems security and privacy (ICISSP)*, 2016, pp. 407–414.

[104] "Botnet dataset," https://www.unb.ca/cic/datasets/botnet.html. last accessed 23 Juin 2019.

[105] E. B. Beigi, H. H. Jazi, N. Stakhanova, and A. A. Ghorbani, "Towards effective feature selection in machine learning-based botnet detection approaches," in *2014 IEEE Conference on Communications and Network Security*. IEEE, 2014, pp. 247–255.

[106] H. Gonzalez, N. Stakhanova, and A. A. Ghorbani, "Droidkin: Lightweight detection of android apps similarity," in *International Conference on Security and Privacy in Communication Networks*. Springer, 2014, pp. 436–453.

[107] "Tor-nontor dataset," https://www.unb.ca/cic/datasets/tor.html. last accessed 30 May 2019.

[108] A. H. Lashkari, G. Draper-Gil, M. S. I. Mamun, and A. A. Ghorbani, "Characterization of tor traffic using time based features." in *ICISSP*, 2017, pp. 253–262.

[109] "Cic dos dataset," https://www.unb.ca/cic/datasets/dos-dataset.html. last accessed 30 May 2019.

[110] H. H. Jazi, H. Gonzalez, N. Stakhanova, and A. A. Ghorbani, "Detecting http-based application layer dos attacks on web servers in the presence of sampling," *Computer Networks*, vol. 121, pp. 25–36, 2017.

[111] G. Szabó, D. Orincsay, S. Malomsoky, and I. Szabó, "On the validation of traffic classification algorithms," in *International Conference on Passive and Active Network Measurement*. Springer, 2008, pp. 72–81.

[112] "Ctu-13 dataset," https://mcfp.weebly.com/the-ctu-13-dataset-a-labeled-dataset-with-botnet-normal-and-background-traffic.html. last accessed 30 May 2019.

[113] S. Garcia, M. Grill, J. Stiborek, and A. Zunino, "An empirical comparison of botnet detection methods," *computers & security*, vol. 45, pp. 100–123, 2014.

[114] "Ssh datasets," https://www.simpleweb.org/wiki/index.php/SSH_datasets. last accessed 30 May 2019.

[115] R. Hofstede, L. Hendriks, A. Sperotto, and A. Pras, "Ssh compromise detection using netflow/ipfix," *ACM SIGCOMM computer communication review*, vol. 44, no. 5, pp. 20–26, 2014.

[116] "Ugr dataset," https://nesg.ugr.es/nesg-ugr16/. last accessed 30 May 2019.

[117] G. Maciá-Fernández, J. Camacho, R. Magán-Carrión, P. García-Teodoro, and R. Theron, "Ugr âĂŸ16: A new dataset for the evaluation of cyclostationarity-based network idss," *Computers & Security*, vol. 73, pp. 411–424, 2018.

[118] "Android malware dataset," https://www.unb.ca/cic/datasets/andmal2017.html. last accessed 30 May 2019.

[119] A. H. Lashkari, A. F. A. Kadir, L. Taheri, and A. A. Ghorbani, "Toward developing a systematic approach to generate benchmark android malware datasets and classification," in *2018 International Carnahan Conference on Security Technology (ICCST)*. IEEE, 2018, pp. 1–7.

[120] "Url dataset," https://www.unb.ca/cic/datasets/url-2016.html. last accessed 30 May 2019.

[121] M. S. I. Mamun, M. A. Rathore, A. H. Lashkari, N. Stakhanova, and A. A. Ghorbani, "Detecting malicious urls using lexical analysis," in *International Conference on Network and System Security*. Springer, 2016, pp. 467–482.

[122] "Bot-iot dataset," https://www.unsw.adfa.edu.au/unsw-canberra-cyber/cybersecurity/ADFA-NB15-Datasets/bot_iot.php. last accessed 30 May 2019.

[123] N. Koroniotis, N. Moustafa, E. Sitnikova, and B. Turnbull, "Towards the development of realistic botnet dataset in the internet of things for network forensic analytics: Bot-iot dataset," *Future Generation Computer Systems*, vol. 100, pp. 779–796, 2019.

[124] "Cse-cic-ids2018 dataset," https://www.unb.ca/cic/datasets/ids-2018.html. last accessed 30 May 2019.

[125] A. Sperotto, R. Sadre, F. Van Vliet, and A. Pras, "A labeled data set for flow-based intrusion detection," in *International Workshop on IP Operations and Management*. Springer, 2009, pp. 39–50.

[126] T. Morris and W. Gao, "Industrial control system traffic data sets for intrusion detection research," in *International Conference on Critical Infrastructure Protection*. Springer, 2014, pp. 65–78.

[127] J. M. Beaver, R. C. Borges-Hink, and M. A. Buckner, "An evaluation of machine learning methods to detect malicious scada communications," in *2013 12th International Conference on Machine Learning and Applications*, vol. 2. IEEE, 2013, pp. 54–59.

[128] T. H. Morris, Z. Thornton, and I. Turnipseed, "Industrial control system simulation and data logging for intrusion detection system research," *7th Annual Southeastern Cyber Security Summit*, pp. 3–4, 2015.

[129] "Android adware dataset," https://www.unb.ca/cic/datasets/android-adware.html. last accessed 30 May 2019.

[130] I. Sharafaldin, A. Habibi Lashkari, and A. A. Ghorbani, "A detailed analysis of the cicids2017 data set," in *Information Systems Security and Privacy*, P. Mori, S. Furnell, and O. Camp, Eds. Cham: Springer International Publishing, 2019, pp. 172–188.

[131] N. Moustafa, J. Slay, and G. Creech, "Novel geometric area analysis technique for anomaly detection using trapezoidal area estimation on large-scale networks," *IEEE Transactions on Big Data*, 2017.

[132] "Cicflowmeter," https://www.unb.ca/cic/research/applications.html#CICFlowMeter. last accessed 30 May 2019.

[133] G. Creech and J. Hu, "Generation of a new ids test dataset: Time to retire the kdd collection," in *2013 IEEE Wireless Communications and Networking Conference (WCNC)*. IEEE, 2013, pp. 4487–4492.

[134] S. Pan, T. Morris, and U. Adhikari, "Classification of disturbances and cyber-attacks in power systems using heterogeneous time-synchronized data," *IEEE Transactions on Industrial Informatics*, vol. 11, no. 3, pp. 650–662, 2015.

[135] T. Morris, A. Srivastava, B. Reaves, W. Gao, K. Pavurapu, and R. Reddi, "A control system testbed to validate critical infrastructure protection concepts," *International Journal of Critical Infrastructure Protection*, vol. 4, no. 2, pp. 88–103, 2011.

[136] G. Bissias, B. N. Levine, M. Liberatore, and S. Prusty, "Forensic identification of anonymous sources in oneswarm," *IEEE Transactions on Dependable and Secure Computing*, vol. 14, no. 6, pp. 620–632, 2015.

[137] A. H. Lashkari, A. F. A. Kadir, H. Gonzalez, K. F. Mbah, and A. A. Ghorbani, "Towards a network-based framework for android malware detection and characterization," in *2017 15th Annual Conference on Privacy, Security and Trust (PST)*. IEEE, 2017, pp. 233–23 309.

[138] L. Deng and D. Yu, "Deep learning: methods and applications," *Foundations and Trends® in Signal Processing*, vol. 7, no. 3–4, pp. 197–387, 2014.

[139] W. Liu, Z. Wang, X. Liu, N. Zeng, Y. Liu, and F. E. Alsaadi, "A survey of deep neural network architectures and their applications," *Neurocomputing*, vol. 234, pp. 11–26, 2017.

[140] J. L. Elman, "Finding structure in time," *Cognitive science*, vol. 14, no. 2, pp. 179–211, 1990.

[141] M. I. Jordan, "Serial order: A parallel distributed processing approach," in *Advances in psychology*. Elsevier, 1997, vol. 121, pp. 471–495.

[142] H. Jaeger and H. Haas, "Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication," *science*, vol. 304, no. 5667, pp. 78–80, 2004.

[143] G. Gelly and J.-L. Gauvain, "Optimization of rnn-based speech activity detection," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 26, no. 3, pp. 646–656, 2017.
[144] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *European conference on computer vision*. Springer, 2014, pp. 818–833.
[145] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1–9.
[146] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.
[147] J. Gu, Z. Wang, J. Kuen, L. Ma, A. Shahroudy, B. Shuai, T. Liu, X. Wang, G. Wang, J. Cai *et al.*, "Recent advances in convolutional neural networks," *Pattern Recognition*, vol. 77, pp. 354–377, 2018.
[148] A. Fischer and C. Igel, "An introduction to restricted boltzmann machines," in *iberoamerican congress on pattern recognition*. Springer, 2012, pp. 14–36.
[149] G. E. Hinton, "Deep belief networks," *Scholarpedia*, vol. 4, no. 5, p. 5947, 2009.
[150] R. Salakhutdinov and H. Larochelle, "Efficient learning of deep boltzmann machines," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, 2010, pp. 693–700.
[151] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," *Journal of machine learning research*, vol. 11, no. Dec, pp. 3371–3408, 2010.